VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

SYSTÉMY L GRAMATIK

L GRAMMAR SYSTEMS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. PETER HNAT**
AUTHOR

VEDOUCÍ PRÁCE **Prof. RNDr. ALEXANDER MEDUNA, CSc.**
SUPERVISOR

BRNO 2018

Zadání diplomové práce/20382/2017/xhnatp00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2017/2018

Zadání diplomové práce

Řešitel: **Hnat Peter, Bc.**
Obor: Počítačová grafika a multimédia
Téma: **Systémy L gramatik**
L Grammar Systems
Kategorie: Teoretická informatika

Pokyny:

1. Dle instrukcí vedoucího se seznamte s gramatickými systémy a L gramatikami.
2. Zaveďte systémy L gramatik analogicky jako gramatické systémy.
3. Studujte vlastnosti systémů L gramatik dle instrukcí vedoucího.
4. Dle pokynů vedoucího uvažujte vhodné aplikace systémů L gramatik, např. v oblasti grafiky. Systémy L gramatik formalizujte.
5. Implementujte formalizace navržené v bodě 4. Zaměřte se na kvalitní vizualizaci implementace.
6. Zhodnoťte dosažené výsledky. Diskutujte další vývoj projektu.

Literatura:

- Rozenberg, G., Salomaa, A. (eds.): Handbook of Formal Languages, Volume 1-3, Springer, 1997, ISBN 3-540-60649-1
- Aho, A.V., Lam, M.S., Sethi, R., Ullman, J.D.: Compilers: Principles, Techniques, and Tools (2nd Edition), Pearson Education, 2006, ISBN 0-321-48681-1

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci dřívějších projektů (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Meduna Alexander, prof. RNDr., CSc.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2017
Datum odevzdání: 23. května 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Diplomová práca sa zaoberá systémami L gramatík. Študované sú základné druhy gramatických systémov. Rozobrané sú rôzne druhy L gramatík a ich použitie v oblasti simulácie rastlinného vývoja. Práca sa zameriava na otvorené L gramatiky. Navrhnutý je paralelný systém zložený z otvorených L gramatík. Komunikácia prebieha prostredníctvom výmeny informácií medzi rastlinami a prostredím. Implementovaný je reálny druh stromu. Simulované je typické prostredie v ktorom sa tento druh stromu bežne vyskytuje. Skúmané sú vlastnosti systému z pohľadu simulácie a časovej náročnosti interpretácie reťazcov. V závere sú uvedené ďalšie možnosti vývoja projektu spolu s možnosťami využitia navrhnutého systému.

Abstract

Master thesis deals with L grammar systems. The basic types of grammar systems are studied. There are analyzed different types of L grammars and their use in field of plant development simulation. The work focuses on open L grammars. A parallel system consisting of open L grammars is designed. Communication takes place through the exchange of information between plants and their environment. Implemented is real kind of tree. Simulated is typical environment in which this kind of tree occurs normally. The system properties are examined from the point of view of simulation and the time-consuming interpretation of strings. At the end of this thesis there are described some other options of future development of the project together with the possibilities of using the proposed system.

Kľúčové slová

Gramatické systémy, otvorené L gramatiky, modelovanie vývoja rastlín, Alstonia Scholaris, JavaScript, React, BabylonJS.

Keywords

Grammar systems, open L grammars, modeling of plant development, Alstonia Scholaris, JavaScript, React, BabylonJS.

Citácia

HNAT, Peter. *Systémy L gramatik*. Brno, 2018. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Prof. RNDr. Alexander Meduna, CSc.

Systémy L gramatik

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením pána Prof. RNDr. Alexanadra Medunu, CSc. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Peter Hnat
21. mája 2018

Poďakovanie

Rád by som poďakoval svojmu vedúcemu Prof. RNDr. Alexandrovi Medunovi, CSc. za všetky cenné rady a usmernenia a za všetok čas a ochotu, ktoré mi venoval počas vypracovávania diplomovej práce.

Obsah

Kapitola 1

Úvod

Rozmanitosť objektov, ktoré sa vyskytujú v prírode je priam nekonečná. L gramatiky boli primárne navrhnuté a prispôsobené na generovanie rôznorodých biologických štruktúr a simulovanie vývoja objektov z reálneho sveta. Nakoľko sú L gramatiky predstaviteľmi procedurálneho generovania, predstavujú výborný nástroj pre vytvorenie množstva vzájomne podobných artefaktov.

Ich zakladateľom bol maďarský biológ Aristid Lindenmayer. Ten tento formalizmus zadefinoval v roku 1968. Neskôr poľský informatik Przemyslaw Prusinkiewicz interpretoval symboly L gramatiky pomocou korytnačej grafiky. Jednotlivé symboly predstavovali grafické elementy ako sú napr. úsečky. Postupom času sa L gramatiky začali používať aj v iných oblastiach. Príkladom je modelovanie toku riek vo fraktálových horách, či ulice vo virtuálnom meste. L gramatiky môžu byť použité aj v iných oboroch než len v počítačovej grafike. Napríklad môžeme pomocou nich generovať hudbu. Známym príkladom je použitie L gramatík vo filme Avatar, kde boli použité na generovanie stromov, rastlín a papradí.

Organizmus je zložený z diskrétnych jednotiek, ktoré označujeme ako moduly. Povaha jednotlivých modulov nie je presne preddefinovaná pomocou formalizmov, čo umožňuje na moduly nahliadať z viacerých strán. V modeloch vyšších rastlín môžu moduly reprezentovať stonku, listy alebo kvety. Každý modul odpovedá symbolu (jeden znak z abecedy L gramatiky). Ten udáva typ modulu. Charakteristiku modulu môžeme ďalej rozšíriť o ďalšie parametre a simulovať tak zložitejšie biologické procesy.

Rozšírením gramatických pravidiel o rôzne podmienky nám L gramatiky umožňujú simulovať rôzne biologické javy ako je rast a s ním spojená distribúcia živín alebo reakcie rastlinného organizmu na okolité prostredie. Ďalším dôležitým faktorom, ktorý umožňuje použiť L gramatiky na simulovanie vývoja rôznych biologických štruktúr je ich paralelný princíp. Všetky časti organizmu sa tak v čase vyvíjajú naraz, čo odpovedá realite. Takisto je možné využiť princípy známe z otvorených L gramatík, pri ktorých dochádza ku komunikácii L gramatík s okolitým prostredím a dovoľujú nám tak simulovať celé ekosystémy.

Každému modulu môžeme priradiť príslušný geometrický význam napr. generovanie objektu, rotáciu alebo inú akciu. Výsledný reťazec je tak interpretovaný korytnačou grafikou. Každý modul vkladáme ako príkaz fiktívnej korytnačky. Tá putuje rovinou a interpretuje modul na základe jeho geometrického významu. Príkazy korytnačky boli rozšírené taktiež pre pohyb v trojdimenzionálnom priestore.

1.1 Cieľ práce

Cieľom práce je naštudovať a zoznámiť sa s gramatickými systémami a rôznymi druhmi L gramatík. Rozobraté sú možnosti ich využitia hlavne v oblasti počítačovej grafiky a simulácie. Na základe získaných znalostí je navrhnutý a formálne zadefinovaný vlastný systém L gramatík so zameraním na interakciu rastlinných organizmov s prostredím, v ktorom sa vyskytujú. Navrhnutý systém je v práci študovaný a sledované sú jeho vlastnosti. Testovaná je najmä výkonnosť systému s ohľadom na grafické vykresľovanie objektov v scéne. V neposlednom rade je cieľom práce naznačiť smery ďalšieho možného vývoja projektu. Záver práce zhrňuje dosiahnuté výsledky.

Kapitola 2 predstavuje úvod do problematiky. Obsahuje formálne definície a opisuje vlastnosti, typické pre bezkontextové gramatiky. Taktiež sú rozobraté základné pojmy potrebné pre pochopenie ďalších častí práce.

Gramatické systémy sú detailne opísané v kapitole 3. Kapitola obsahuje základné delenie gramatických systémov. Detailne popisuje paralelné gramatické systémy na ktorých bude založený aj navrhnutý systém L gramatík.

Kapitola 4 obsahuje detailný popis a formálne definície rôznych druhov L gramatík. Rozobrané sú gramatiky s kontextovou podmienkou, ktoré umožňujú simulovať biologické javy zavedením podmienok do gramatických pravidiel. Záver kapitoly sa zameriava na reálne využitie L gramatík v oblasti počítačovej grafiky. Ďalej sú skúmané vlastnosti L gramatík, ktoré pracujú paralelne a bezkontextových gramatík, ktoré pristupujú ku generovaní reťazca sekvenčným spôsobom.

Každá kapitola obsahuje okrem formálnych definícií radu príkladov a názorných ukážok pre lepšie preniknutie do danej problematiky.

Kapitola 5 obsahuje návrh implementovaného systému L gramatík. Ide o formálne definície a návrh implementácie. Pri návrhu aplikácie bol kladený dôraz na použitie modelu rastlinného organizmu z reálneho sveta. Konkrétne bol použitý model stromu typický pre oblasť juhovýchodnej Ázie. Tento strom je v latinčine známy pod názvom *Alstonia Scholaris*. K modelu stromu bol následne vytvorený model prostredia, ktorý odpovedá životnému prostrediu typickému pre danú oblasť sveta. Detaily použitých modelov spolu s hlavnými stavebnými blokmi programu obsahuje spomínaná časť práce.

Implementácia obsahuje popis riešenia zaujímavých častí programu. Podrobne sú rozobraté dátové štruktúry použité pri interpretácii vygenerovaného reťazca. Nakoľko je v aplikácii potrebná interakcia užívateľa s programom, obsahuje táto časť aj stručný popis užívateľského rozhrania. V programe bol kladený dôraz na kvalitnú grafickú vizualizáciu. Použité knižnice boli vyberané s ohľadom na jednoduchú prenositeľnosť a ľahkú spustiteľnosť finálnej aplikácie. Popis použitých technológií je taktiež obsahom kapitoly 6.

Kapitola 7 obsahuje výsledky získané testovaním aplikácie. Informácie uvedené v tejto časti mali značný vplyv na nadväzujúcu kapitolu 8. Tá je venovaná možným vylepšeniam programu a ďalšiemu vývoju projektu.

V záverečnej kapitole 9 sú zhrnuté výsledky získané v predchádzajúcich častiach.

Kapitola 2

Základné modely pre bezkontextové jazyky

Táto kapitola sa venuje základným formálnym modelom pre popis bezkontextových jazykov. Práve bezkontextové gramatiky sú základom L gramatík, ktorými sa zaoberá celá práca. V tejto časti práce je detailne rozobraný model bezkontextových gramatík z pohľadu generátorov bezkontextových jazykov. Čitateľovi sú priblížené základné vlastnosti príznačné pre túto triedu formálnych jazykov. Objasnené sú základné zložky bezkontextových gramatík a zadefinovaný jazyk, ktorý tieto gramatiky generujú. Kapitola taktiež objasňuje základné pojmy ako *axióm*, *veta* alebo *derivácia* bez ktorých sa ani v ďalších častiach práce nezaobídeme.

Medzi základné modely bezkontextových jazykov patria *zásobníkové automaty* z *angl. Pushdown automata* a *bezkontextové gramatiky* z *angl. Context–free grammars*. Na rozdiel od zásobníkových automatov, ktoré sú akceptormi bezkontextových jazykov sú gramatiky predstaviteľmi generátorov. Bezkontextové gramatiky teda predstavujú jazyk-generujúci prepisovací systém. Každé z prepisovacích pravidiel gramatiky obsahuje jeden symbol na ľavej strane. Opakovaným aplikovaním prepisovacích pravidiel je gramatika schopná generovať vety, ktoré patria do jej jazyka [21].

2.1 Bezkontextové gramatiky

Bezkontextové gramatiky teda predstavujú jazyk-generujúce systémy založené na konečnom počte terminálnych symbolov, neterminálnych symbolov a gramatických pravidiel. Terminálne symboly sa na rozdiel od neterminálnych symbolov vyskytujú v jazyku, ktorý daná bezkontextová gramatika generuje. Pre každé pravidlo platí, že jeho ľavá strana je tvorená jedným neterminálnym symbolom. Na pravej strane pravidla sa potom nachádza reťazec, ktorý je tvorený terminálnymi aj neterminálnymi symbolmi. Gramatika začína generovať reťazec z počiatočného neterminálu. Ten je označovaný ako *štartovací neterminál*. Z počiatočného neterminálu následne gramatika aplikovaním jednotlivých pravidiel prepisuje neterminálne symboly vzhľadom na aplikované pravidlá dovtedy, pokiaľ sa nedopracuje k vete. *Veta* v tomto prípade predstavuje reťazec tvorený len terminálnymi symbolmi. Množina všetkých viet reprezentuje *jazyk* generovaný gramatikou.

Bezkontextovú gramatiku teda tvoria štyri základné komponenty [1]:

- Množina *terminálnych* symbolov. Terminály sú základné symboly jazyka definované gramatikou.

- Množina *neterminálnych* symbolov, tzv. *syntaktické premenné*. Každý neterminál predstavuje množinu terminálnych reťazcov.
- Množinu *prepisovacích pravidiel*, kde každé pravidlo pozostáva z neterminálu tzv. *hlava* resp. *ľavá* strana pravidla, symbolu šípky a postupnosti terminálov a neterminálov, ktoré nazývame *telo* resp. *pravá* strana pravidla.
- Označenie jedného z neterminálnych symbolov ako *štartovací neterminál*.

Nakoľko bezkontextové gramatiky sú zástupcami formálnych gramatík s frázovou štruktúrou, formálne si najprv zadefinujeme tie. Následne nadviažeme na ich formálnu definíciu a špecifikujeme ju pre bezkontextové gramatiky.

2.2 Gramatiky s frázovou štruktúrou

Definícia 2.1. *Gramatika s frázovou štruktúrou.* Gramatika s frázovou štruktúrou je štvorica $G = (V, T, P, S)$, kde

- V je úplná *abeceda*,
- T je množina *terminálov* ($T \subset V$),
- $P \subseteq V^*(V - T)V^* \times V^*$ je *konečná relácia* (množina pravidiel),
- $S \in V - T$ je *axióm* z G (štartovací neterminál).

Symboly z množiny, ktorá vznikne z $V - T$ označujeme ako *neterminály*. Dvojicu $(x, y) \in P$ nazývame *pravidlo*. Formálne zapisujeme pravidlo nasledovne:

$$x \to y \in P.$$

Jednotlivé pravidlá tvoria množinu pravidiel P, ktorá obsahuje všetky prepisovacie pravidlá danej gramatiky G.

Definícia 2.2. *Priama derivácia.* Relácia priamej derivácie v G je binárna relácia nad množinou V^*, ktorú značíme ako $\Rightarrow_G$ a definujeme nasledovne. Nech $x \to y \in P, u, v, z_1, z_2 \in V^*$, a $u = z_1xz_2$, $v = z_1yz_2$; potom,

$$u \Rightarrow_G v[x \to y].$$

Tento zápis môžeme zjednodušiť a namiesto $u \Rightarrow_G v[x \to y]$, používať iba $u \Rightarrow_G v$. Zjednodušene povedané dva reťazce u a v sú v relácii priamej derivácie vtedy, ak aplikovaním pravidla $x \to y$ sa z u doderivujeme do v. V závislosti na počte krokov, v ktorých sa doderivujeme aplikovaním jednotlivých pravidiel do daného reťazca, definujeme *tranzitívny*, značíme ako $\Rightarrow^+$, a *tranzitívno reflexivný* uzáver priamej derivácie $\Rightarrow_G$, ktorý značíme ako $\Rightarrow^*$. Ak $S \Rightarrow^*_G$ pre nejaké $x \in V^*$, x nazývame *vetnou formou*.

Definícia 2.3. *Jazyk gramatiky s frázovou štruktúrou.* Jazyk gramatiky G, značíme ako $L(G)$, je formálne definovaný ako:

$$L(G) = \{w \in T^* \mid S \Rightarrow^*_G w\}.$$

Ide teda o množinu všetkých terminálnych reťazcov ku ktorým sme sa schopní doderivovať z počiatočného neterminálu. Pre tieto gramatiky je ďalej príznačné, že sú generátormi rekurzívne vyčísliteľných jazykov, ktoré označujeme RE. Bezkontextové gramatiky sú predstaviteľmi gramatík s frázovou štruktúrou.

Definícia 2.4. *Bezkontextová gramatika.* Bezkontextová gramatika G je štvorica $G = (V, T, P, S)$, kde každé pravidlo je v tvare $x \to y \in P$ a $x \in V - T$. *Bezkontextový jazyk* je jazyk generovaný bezkontextovou gramatikou.

Definície uvedené v tejto podkapitole sú prevzaté z [22].

Kapitola 3

Gramatické systémy

Gramatiky a automaty sú predstaviteľmi klasických výpočtových prostriedkov, ktoré sú detailne popísané v teórii formálnych jazykov a automatov. V klasickej teórii formálnych jazykov je jazyk generovaný jednou gramatikou resp. prijímaný jedným automatom. Novodobá počítačová veda sa však zameriava na distribúciu týchto výpočtov medzi viacero formálnych prostriedkov. Práve to podmienilo vznik teórie gramatických systémov. Táto veda vymedzuje základné pojmy tejto problematiky ako *distribúcia*, *paralelizmus* a *komunikácia*, ktoré budú podrobnejšie rozobrané v nasledujúcich podkapitolách. Najprv je však potrebné zadefinovať samotné gramatické systémy. V jednoduchosti je *gramatický systém* z *angl. Grammar system* tvorený množinou gramatík, ktoré spolupracujú podľa špecifikovaného protokolu a generujú jeden jazyk. Najdôležitejší prvok tohto procesu je práve *protokol*, ktorým je definovaná spolupráca jednotlivých častí systému. Tento fakt nám umožňuje na teóriu gramatických systémov nahliadať aj ako na gramatickú teóriu spolupracujúcich protokolov. Informácie uvedené v tejto kapitole sú prevzaté z [30].

3.1 Základné rozdelenie

Gramatické systémy môžeme rozdeliť na dve základné triedy:

1. Sekvenčné.
2. Paralelné.

Spolupracujúce distribuované gramatické systémy z *angl. Coperating distributed grammar systems* patria medzi *sekvenčné* gramatické systémy. Pre tieto systémy platí, že všetky zložkové gramatiky majú *spoločnú vetnú formu*. V jeden moment je aktívna len jedna gramatika, ktorá prepisuje aktuálnu vetnú formu. *Protokol spolupráce* presne vymedzuje to, ktorá časť systému bude aktívna v daný moment a kedy aktívna gramatika skončí, teda kedy aktuálnu vetnú formu začne prepisovať iná gramatika. Základnými ukončovacími podmienkami sú napríklad, že aktívna komponenta pracuje *presne k krokov*, *najmenej k krokov*, *najviac k krokov* alebo *maximálne k krokov* (za krok v tomto ponímaní považujeme aplikáciu prepisovacieho pravidla). Samozrejme môžeme použiť aj ďalšie ukončovacie podmienky v závislosti na činnosti použitého gramatického systému. Súbor terminálnych reťazcov generovaných týmto gramatickým systémom nazývame *generovaný jazyk*.

Pre *paralelne komunikujúce gramatické systémy* z *angl. Parallel communicating grammar systems* platí, že každá komponenta má *vlastnú vetnú formu*. V rámci každej časovej

jednotky, každá komponenta aplikuje pravidlo a prepíše svoju vlastnú vetnú formu. Kľúčovým prvkom týchto gramatických systémov je *mechanizmus komunikácie pomocou dotazov*. Špeciálne symboly sú vybavené každým symbolom smerujúcim na komponentu systému. Ak komponenta i vygeneruje dotazovací symbol Q_j, potom aktuálna vetná forma komponenty j bude poslaná komponente i, pričom v odoslanej vetnej forme budú nahradené všetky výskyty symbolu Q_j. Jedna komponenta systému je navrhnutá ako *master*. *Jazyk generovaný touto komponentou* je zároveň jazykom celého systému. Samozrejme môžeme uvažovať aj iné komunikačné mechanizmy.

3.2 Model tabule

Štruktúra modelu tzv. *tabule* z *angl. Blackboard* sa používa v oblasti riešenia problémov v sekvenčných gramatických systémoch. V týchto systémoch považujeme za tabuľu *spoločnú vetnú formu*. Táto spoločná dátová štruktúra v sebe uchováva *aktuálny stav problému*, ktorý je riešený. Jednotlivé gramatiky predstavujú *vedomostné zdroje,* známe aj ako *agenti*, *procedúry* alebo *procesory*, ktoré prispievajú k riešeniu problému zmenou obsahu tabule. *Protokol spolupráce* kóduje kontrolu jednotlivých vedomostných zdrojov.

V paralelných gramatických systémoch sa používa *modifikovaná verzia* modelu tabule. Pre tento model je typické, že každý *agent* má svoj vlastný *poznámkový blok* z *angl. Notebook*, ktorý obsahuje popis čiastočného podproblému zadaného problému, ktorý má riešiť celá skupina agentov. Každý agent spravuje svoj vlastný poznámkový blok. Jeden významný agent spravuje tabuľu. Tento významný agent má na rozdiel od ďalších agentov k dispozícii popis celého zadaného problému. Taktiež rozhoduje kedy môžeme problém považovať za vyriešený. Celý tento koncept poznáme pod názvom *triedny model* z *angl. the Classroom model* riešenia problému. *Triedny líder* z *angl. Master* operuje nad tabuľou. *Žiaci* z *angl. Pupils* (ostatní agenti) majú za úlohu riešiť čiastočný problém za pomoci vlastných *poznámkových blokov*. Žiaci medzi sebou môžu komunikovať buď to *v každom kroku riešenia problému* alebo len na *vyžiadanie triedneho lídra*. K vyriešeniu globálneho problému dochádza na základe kooperácie nad tabuľou.

3.3 Paralelné gramatické systémy

Definícia 3.1. *Paralelný gramatický systém. Paralelný gramatický systém* stupňa $n, n \geq 1$, je $n+3$–tica $\Gamma = (N, K, T, (S_1, P_1), \ldots, (S_n, P_n))$, kde

- N abeceda *neterminálnych* symbolov,
- T abeceda *terminálnych* symbolov,
- $K = \{Q_1, Q_2, \ldots, Q_n\}$,
- P_i je konečná *množina prepisovacích pravidiel* nad $N \cup K \cup T$, pre každé i, kde $1 \leq i \leq n$,
- $S_i \in N$ je *štartovací neterminál*, pre každé i, kde $1 \leq i \leq n$.

Nech $V_\Gamma = N \cup K \cup T$. Množiny prepisovacích pravidiel P_i, $1 \leq i \leq n$, nazývame *komponentami* systému. Prvky $Q_1, \ldots, Q_n$ z množiny K sú *dotazovacie symboly* z *angl. Query symbols*. Index i z Q_i, ukazuje na i–tú komponentu P_i gramatického systému.

V prípade, že chceme ako komponenty paralelného gramatického systému Γ použiť gramatiky, takýto gramatický systém zapíšeme ako $\Gamma = (N, K, T, G_1, \ldots, G_n)$ s $G_i = (N \cup K, T, S_i, P_i)$, $1 \leq i \leq n$. Tak ako v prípade sekvenčných gramatických systémov môžeme použiť ako komponenty nezávislé gramatiky. N–ticu $(x_1, x_2 \ldots, x_n)$ s $x_i \in V_\Gamma^*$ pre každé i, $1 \leq i \leq n$, nazývame *konfiguráciou systému* Γ.

Definícia 3.2. *Derivácia v paralelnom gramatickom systéme. Paralelný gramatický systém* $\Gamma = (N, K, T, (S_1, P_1), \ldots, (S_n, P_n))$, pre dve n–tice $(x_1, x_2 \ldots, x_n)$, $(y_1, y_2 \ldots, y_n)$, s $x_i, y_i \in V_\Gamma^*$, $1 \leq i \leq n$, kde $x_1 \notin T^*$, píšeme $(x_1, x_2 \ldots, x_n) \Rightarrow (y_1, y_2 \ldots, y_n)$ vtedy, ak je splnená jedna z nasledujúcich podmienok:

1. Pre každé $i, 1 \leq i \leq n$, $|x_i|_K = 0, 1 \leq i \leq n$ a pre každé $i, 1 \leq i \leq n$, máme buď $x_i \Rightarrow y_i$ na základe pravidla z P_i alebo $x_i = y_i \in T^*$.

2. Alebo existuje i, $1 \leq i \leq n$, také, že $|x_i|_K > 0$. Nech pre každé $i, x_i = z_1 Q_{i_1} z_2 Q_{i_2} \ldots z_t Q_{i_t} z_{t+1}, t \geq 1$, pre $z_j \in (N \cup T)^*, 1 \leq j \leq t+1$. Ak $|x_i|_K = 0$ pre každé $j, 1 \leq j \leq t$, potom $y_i = z_1 x_{i_1} z_2 x_{i_2} \ldots z_t x_{i_t} z_{t+1}$ a $y_{i_j} = S_{i_j}, 1 \leq j \leq t$. Ak pre nejaké $j, 1 \leq j \leq t, |x_i|_K \neq 0$, potom $y_i = x_i$. Pre každé $i, 1 \leq i \leq n$, ak y_i nieje špecifikované, platí $y_i = x_i$.

Z konfigurácie $(x_1, x_2 \ldots, x_n)$ systém prejde priamo do konfigurácie $(y_1, y_2 \ldots, y_n)$ vtedy, ak:

1. Žiaden z dotazovacích symbolov sa nevyskytuje v $x_1, x_2 \ldots, x_n$, takže máme deriváciu $x_i \Rightarrow y_i$, v každej komponente P_i, $1 \leq i \leq n$ je použité jedno pravidlo, okrem prípadu, kedy je x_i terminálny symbol, $x_i \in T^*$. V takomto prípade je $y_i = x_i$.

2. V prípade výskytu dotazovacích symbolov v niektorom z x_i je *komunikačný krok* realizovaný nasledovne. Každý výskyt dotazovacieho symbolu Q_j v x_i je nahradený x_j, pričom x_j už neobsahuje žiadne dotazovacie symboly. Presnejšie, komponenta x_i (ktorá obsahuje dotazovacie symboly) je upravená len vtedy, ak všetky dotazovacie symboly, ktoré sa v nej vyskytujú, ukazujú na reťazce, ktoré už neobsahujú dotazovacie symboly. Po tomto kroku, gramatika G_j pokračuje v prepisovaní od svojho axiómu. K prepisovaniu nedochádza tak dlho, pokiaľ je prítomný čo i len jeden dotazovací symbol. V prípade, že niektoré z dotazovacích symbolov nie sú nahradené v danom kroku, môžu byť nahradené v kroku nasledujúcom. Musíme taktiež poznamenať, že pravidlá v tvare $x_1 Q_i x_2 \to x$ nie sú nikdy použité, nakoľko v paralelných gramatických systémoch, ktoré sme si zadefinovali vyššie, takéto pravidlá neuvažujeme. Takisto treba podotknúť, že $(x_1, \ldots, x_n) \Rightarrow (y_1, \ldots, y_n)$ nie je definované v prípade, že $x_1 \in T^*$.

Definícia 3.3. *Jazyk generovaný paralelným gramatickým systémom. Jazyk generovaný paralelným gramatickým systémom* Γ je definovaný ako

$$L(\Gamma) = \{x \in T^* \mid (S_1, S_2, \ldots, S_n) \Rightarrow^* (x, \alpha_2, \ldots, \alpha_n), \alpha_i \in V_\Gamma^*, 2 \leq i \leq n\}.$$

Začíname teda z n–tice axiómov $(S_1, S_2, \ldots, S_n)$ a opakovaným uskutočňovaním prepisovacích a komunikačných krokov sa chceme dopracovať k stavu, kým komponenta P_1 nebude obsahovať reťazec zložený len z terminálnych reťazcov. Ostatné komponenty nemusia mať vygenerovaný reťazec zložený len z terminálnych symbolov. Akonáhle skončí komponenta P_1, celý gramatický systém končí. Komponentu P_1 teda označujeme ako hlavnú.

Paralelné gramatické systémy môžeme ďalej klasifikovať na základe toho, či sa po uskutočnení komunikačného kroku *vracajú k počiatočnému axiómu* alebo *pokračujú v spracovaní*

aktuálneho reťazca. Paralelný gramatický systém sa vracia k počiatočnému axiómu v prípade, že po komunikačnom kroku sa každá komponenta systému, ktorá odoslala svoj reťazec do inej komponenty, vráti na začiatok a začína generovať reťazec od počiatočného axiómu. V opačnom prípade hovoríme o paralelnom gramatickom systéme bez návratu k počiatočnému axiómu a teda komponenty, ktoré odoslali svoj reťazec do inej komponenty, pokračujú v generovaní aktuálneho reťazca.

Príklad 3.3.1. Uvažujme nasledujúci paralelný gramatický systém:

$$\begin{aligned}
&\Gamma_1 = (\{S_1, S_1', S_2, S_3\}, K, \{a, b\}, (S_1, P_1), (S_2, P_2), (S_3, P_3)),\\
&\quad P_1 = \{S_1 \to abc, S_1 \to a^2b^2c^2, S_1 \to aS_1', S_1 \to a^3Q_2, S_1' \to aS_1', S_1' \to a^3Q_2,\\
&\qquad S_2 \to b^3Q_3, S_3 \to c\},\\
&\quad P_2 = \{S_2 \to bS_2\},\\
&\quad P_3 = \{S_3 \to cS_3\}.
\end{aligned}$$

Potom jazyk generovaný týmto gramatickým systémom je definovaný ako:

$$L_r(\Gamma) = L_{nr}(\Gamma_1) = \{a^n b^n c^n \mid n \geq 1\}.$$

Systém začína z počiatočných axiómov (S_1, S_2, S_3). Aplikovaním pravidla $S_1 \to aS_1'$ z množiny P_1 a následným aplikovaním piateho pravidla $S_1' \to aS_1'$ a jednoznačných pravidiel z množín P_2, P_3 pre $n \geq 0$ krokov, dostávame:

$$(S_1, S_2, S_3) \Rightarrow_r (aS_1', bS_2, cS_3) \Rightarrow_r^* (a^{n+1}S_1', b^{n+1}S_2, c^{n+1}S_3).$$

Nakoniec bude použité šieste pravidlo z množiny P_1:

$$(a^{n+1}S_1', b^{n+1}S_2, c^{n+1}S_3) \Rightarrow_r (a^{n+4}Q_2, b^{n+2}S_2, c^{n+2}S_3).$$

Nakoľko sa v reťazci vyskytuje dotazovací symbol Q_2, je potrebné vykonať komunikačný krok: reťazec $b^{n+2}S_2$ je poslaný prvej komponente, kde nahradí dotazovací symbol Q_2.

$$(a^{n+4}Q_2, b^{n+2}S_2, c^{n+2}S_3) \Rightarrow_r (a^{n+4}b^{n+2}S_2, S_2, c^{n+2}S_3)$$

.

Z príkladu 3.3.1 je jasné, že uvedený paralelný gramatický systém Γ_1 sa vracia k počiatočnému axiómu potom, čo bol dokončený komunikačný krok. Konkrétne ide o komponentu 2, ktorá opäť začína z počiatočného axiómu S_2 a to hneď potom, čo odoslala svoj reťazec b^{n+2} do prvej komponenty. Podrobnejší popis činnosti spolu s ďalšími prikladmi je možné nájsť v literatúre [30].

Kapitola 4

L gramatiky

Každý objekt, ktorý človek vníma vo svojom okolí je unikátny a len ťažko nájdeme dve veci na svete, ktoré by boli na vlas identické. Práve to spôsobuje nekonečnú rozmanitosť flóry a fauny všade navôkol. *L gramatiky* z *angl. L grammars* ako formálne gramatiky boli navrhnuté a prispôsobené na generovanie rozmanitých biologických štruktúr a simulovanie vývoja objektov z reálneho sveta. Primárnym zameraním L gramatík bolo generovanie rastlín všetkého druhu. Vegetácia sa vo voľnej prírode vyskytuje v rôznych formách ako trávy, stromy, kríky a mnoho ďalších. Práve *procedurálne modelovanie* predstavuje výborný nástroj pre vytvorenie obrovského množstva podobných artefaktov. Jednotlivé vygenerované časti sú si navzájom podobné, nie sú však identické, čo presne odpovedá realite. Tento fakt podmienil využitie procedurálneho generovania v oblasti počítačovej grafiky [35].

L gramatiky predstavil A. Lindenmayer ako matematický formalizmus pre modelovanie mnohobunkových organizmov, ktoré tvoria lineárne alebo rozvetvené štruktúry [27]. Celý organizmus je zložený z diskrétnych jednotiek, ktoré spoločne označujeme ako *moduly*. Povaha jednotlivých modulov nie je presne preddefinovaná pomocou formalizmov. Napríklad moduly v nižších jednobunkových organizmoch predstavujú jednotlivé *bunky*. V modeloch vyšších rastlín, môžu moduly reprezentovať *stonku*, *listy* alebo *kvety*. Každý modul teda predstavuje *symbol* (znak z abecedy L gramatiky), ktorý presne špecifikuje *typ* modulu. Okrem toho môžeme charakteristiku modulu doplniť aj o ďalšie číselné parametre, ktoré spolu so symbolom určujú jeho *stav*. Takýto koncept L gramatík poznáme pod názvom *parametrické L gramatiky* z *angl. Parametric L grammars*, ktoré budú rozobrané v tejto kapitole.

L gramatika predstavuje *dynamický model*, čo znamená, že na formu organizmu je nahliadané ako na výsledok jeho vývoja. Ide teda o udalosť v časopriestore, nie iba o konfiguráciu v priestore. Vývoj štruktúry je popísaný pomocou prepisovacích pravidiel, ktoré menia stav modulu. Pomocou jednotlivých prepisovacích pravidiel sme schopní modul ľahko odstrániť alebo ho nahradiť jedným, či viacerými inými modulmi. Prepisovacie pravidlá sú aplikované *paralelne*. Ide teda o súčasné zachytenie priebehu času vo všetkých častiach rastúceho organizmu. L gramatiky sú často prezentované formálnym spôsobom. V tejto kapitole na ne však budeme nahliadať aj v kontexte modelovania. Práve tento pohľad nám umožní vidieť vzťah medzi prírodou a jednotlivými modelmi L gramatík, zdôrazňujúc ich koncepčnú podstatu. Popísané budú rôzne príklady z reálneho sveta, ktoré riešia otázku riadenia času v simuláciách, čo je nevyhnutné pre animáciu vývoja organizmu v čase.

4.1 Model organizmu: Anabaena Catenula

Anabeana [18] predstavuje rod vláknitých siníc, ktoré sú zástupcami mnohobunkových organizmov. Ako aj ostatné cyanobaktérie sú schopné aeróbnej fotosyntézy a fixácie dusíka z atmosféry. Tieto dva rozličné procesy sa vzájomne vylučujú, nakoľko enzýmová dusíkáza zodpovedná za fixáciu dusíka, je ničená kyslíkom, ktorý vzniká pri fotosyntéze. Väčšina cyanobaktérií sa s týmto javom vysporiada tak, že fotosyntéza prebieha cez deň a fixácia dusíka v noci. Taktiež dochádza k rozdeleniu týchto dvoch procesov v priestore. Sinice obsahujú vegetatívne bunky v ktorých prebieha fotosyntéza a heterocysty, ktoré viažu dusík. Jednotlivé bunky sú následne usporiadané do vláken. Tento model sa stal východiskovým modelom organizmu z ktorého vychádzajú nasledujúce podkapitoly.

4.2 Bezkontextové L–gramatiky

Mitchinson a Wilcox [23] chceli vysvetliť pozorovaný model dlhých a krátkych buniek vo vegetatívnom segmente *Anabaeana* ako priamy výsledok vývojového procesu. Pre tento účel boli bunky tvoriace jednotlivé vlákna rozdelené na dve triedy: *dlhé bunky* označované ako L a *krátke bunky* označené pomocou písmena S. Ďalej predpokladali, že každá vegetatívna bunka má jednu z dvoch možných polarít vyznačenú šípkou v hornej oblasti: $\overrightarrow{S}$, $\overrightarrow{L}$, $\overleftarrow{S}$ a $\overleftarrow{L}$.

Počas vývoja sa krátke bunky S predlžujú a menia svoj stav na bunky L. Dlhé bunky L sa delia na jednu bunku L a S. Lindenmayer v svojej práci [19] použil tento model, zohľadňujúc polaritu jednotlivých buniek a zapísal tento proces pomocou nasledujúcej množiny prepisovacích pravidiel:

$$\overrightarrow{S} \to \overrightarrow{L} \qquad \overleftarrow{S} \to \overleftarrow{L} \qquad \overrightarrow{L} \to \overleftarrow{L}\,\overrightarrow{S} \qquad \overleftarrow{L} \to \overleftarrow{S}\,\overrightarrow{L} \tag{4.1}$$

Takéto pravidlá sú základným kameňom tzv. *0L–gramatík*, najjednoduchšieho typu L gramatík. Skratka *d0L* označuje *deterministické L gramatiky s 0 interakciami*, ktoré si formálne zadefinujeme [17].

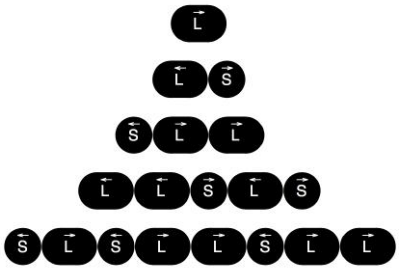
4.2.1 Deterministické L gramatiky (d0L)

Definícia 4.1. *0L gramatika.* Nech V označuje abecedu, V^* množinu všetkých reťazcov nad abecedou V a V^+ množinu všetkých neprázdnych reťazcov nad abecedou V. Potom *0L gramatika* je usporiadaná trojica $G = \langle V, \omega, P \rangle$, kde

- V je *abeceda symbolov*,
- $\omega \in V^+$ je neprázdna postupnosť symbolov zvaná *Axiom*,
- $P \subset V \times V^*$ je *konečná množina prepisovacích pravidiel.*

Pravidlo $(a, \chi) \in P$ píšeme ako $a \to \chi$. Znak a nazývame *predchodca* (ľavá strana), z *angl. Predecessor* a reťazec χ nazývame *následník* (pravá strana), z *angl. Successor* pravidla. Predpokladáme, že pre každé $a \in V$ existuje alespoň jeden následník $\chi \in V^*$ taký, že $a \to \chi$. Pokiaľ nieje pre daného predchodcu $a \in V$ žiadne pravidlo explicitne definované, predpokladá sa, že *identické pravidlo* $a \to a$ patrí do množiny pravidiel P.

Definícia 4.2. *Deterministická 0L gramatika. 0L gramatika* je *deterministická* (označujeme d0L gramatika) vtedy a len vtedy, ak pre každé $a \in V$ existujé práve jedno $\chi \in V^*$ také, že $a \to \chi$.



Obr. 4.1: Vývojový proces delenia krátkych a dlhých buniek *Anabeana*.

Definícia 4.3. *Propagujúca 0L gramatika.* Špeciálnym prípadom 0L gramatiky je *propagujúci 0L gramatika* (označujeme *P0L*, resp. *Pd0L* ak je *deterministická*), ktorá pre žiadne $a \in V$ neobsahuje pravidlo v tvare $a \to \epsilon$, čo znamená, že symboly môžu byť prepisované, ale nesmú byť mazané.

Definícia 4.4. *Derivačný krok.* Nech $\mu = a_1 \dots a_n$ je ľubovoľný reťazec nad abecedou V. Reťazec $v = \chi_1 \dots \chi_m \in V^*$ je *priamo derivovaný* (generovaný) z μ, zapisujeme $\mu \Rightarrow v$ vtedy a len vtedy ak $a_i \to \chi_i$ pre všetky $i = 1, \dots, m$. Reťazec v je generovaný z G deriváciou dĺžky n pokiaľ existuje *postupnosť reťazcov* $\mu_0, \mu_1, \dots, \mu_n$ také, že $\mu_0 = \omega$, $\mu_n = v$ a $\mu_0 \Rightarrow \mu_1 \Rightarrow \dots \Rightarrow \mu_n$.

Definícia 4.5. *0L jazyky. Jazyk* generovaný 0L gramatikou G, značíme $L(G)$, obsahuje všetky reťazce generované z axiómu v ľubovoľnom počte derivačných krokov. Teda $L(G) = \{w \,|\, \omega \Rightarrow^* w\}$.

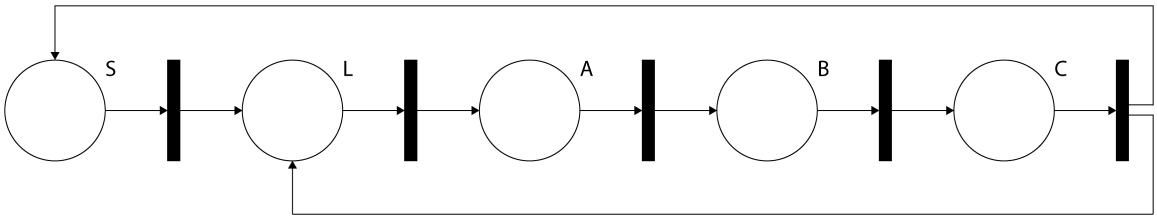
Tak ako všetky L gramatiky aj *0L gramatiky* pracujú nad postupnosťou symbolov, ktoré označujeme ako *reťazce* alebo *slová*. V jednom derivačnom kroku je symbol ľavej strany pravidla nahradený jeho pravou stranou. Vývojový proces je simulovaný ako postupnosť derivačných krokov, počínajúc s počiatočným reťazcom, ktorý nazývame *axióm* a značíme pomocou písmena gréckej abecedy ω. Obrázok 4.1 ilustruje vývojovú postupnosť generovanú L gramatikou $G = \langle V, \omega, P \rangle$ s abecedou $V = \{\overleftarrow{L}, \overrightarrow{L}, \overleftarrow{S}, \overrightarrow{S}\}$. Axióm z ktorého L gramatika začína generovať $\omega = \overrightarrow{L}$, množinu prepisovacích pravidiel P tvoria pravidlá definované v rovnici 4.1.

Ako si môžeme všimnúť, uvedené pravidlá sú *bezkontextové* a spĺňajú tak podmienku *Chomského klasifikácie jazykov* pre bezkontextové gramatiky. Jedná sa teda o jednoduchú bezkontextovú L gramatiku. Medzi Chomského gramatikami a L gramatikami je však zásadný rozdiel, ktorý definuje Veta 1.

Veta 1. V chomského gramatikách dochádza k aplikácii prepisovacích pravidiel *sekvenčne*, v L gramatikách sú prepisovacie pravidlá aplikované *paralelne* pre prepísanie všetkých znakov v danom reťazci.

Tento fakt je kľúčový a odráža biologické zameranie L gramatík. Aj v modeli mnohobunkového organizmu siníc *Anabaena Catenula* spomínaného vyššie, dochádza k deleniu buniek súčasne, teda pri simulácii je potrebné prepisovacie pravidlá aplikovať súčasne, čo presne odpovedá tvrdeniu popísaného vo Vete 1. Paralelné aplikovanie pravidiel má podstatný vplyv aj na vlastnosti prepisovacích systémov.

Veta 2. *Vzťah bezkontextových gramatík a L gramatík.* Existujú jazyky, ktoré môžu byť generované bezkontextovými L gramatikami (0L jazyky), ale nemôžu byť generované bezkontextovými gramatikami v Chomského hierarchii, viď. obrázok 4.4. Tvrdenia uvedené vo Vete 1 a 2 spolu s ilustračným obrázkom 4.4 boli prevzaté z práce [17].

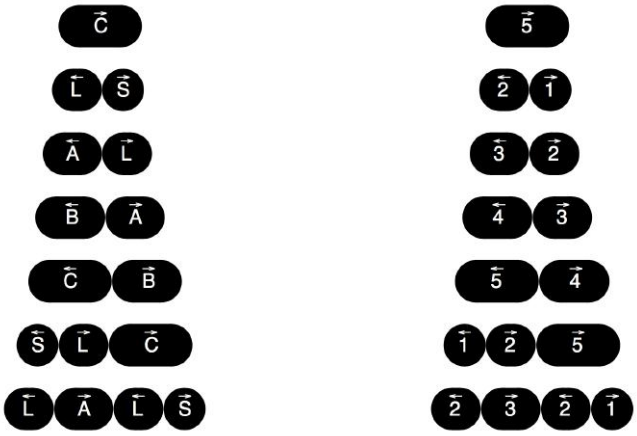


Obr. 4.2: Popis L gramatiky pomocou Petriho siete.

Kvôli kontrole časovania bunkového delenia si dovolíme navrhnutú L gramatiku skomplikovať a jej množinu prepisovacích pravidiel upraviť nasledovne:

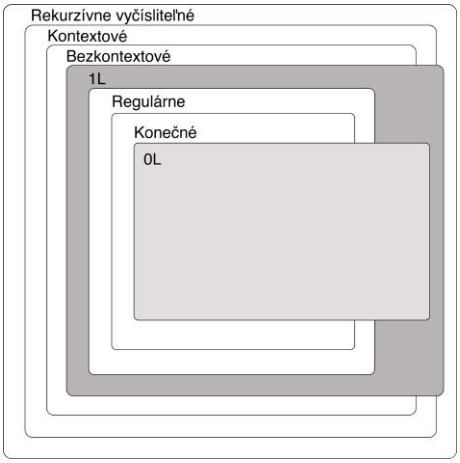
$$\begin{array}{ccccc} \overrightarrow{S} \to \overrightarrow{L} & \overrightarrow{L} \to \overrightarrow{A} & \overrightarrow{A} \to \overrightarrow{B} & \overrightarrow{B} \to \overrightarrow{C} & \overrightarrow{C} \to \overleftarrow{L}\,\overrightarrow{S} \\ \overleftarrow{S} \to \overleftarrow{L} & \overleftarrow{L} \to \overleftarrow{A} & \overleftarrow{A} \to \overleftarrow{B} & \overleftarrow{B} \to \overleftarrow{C} & \overrightarrow{C} \to \overleftarrow{S}\,\overrightarrow{L} \end{array} \tag{4.2}$$

Sekvencie prechodov medzi stavmi modelu môžeme pre uvedenú L gramatiku zakresliť pomocou Petriho siete [26]. Tá je ilustrovaná na obrázku 4.2. Posledný stav C reprezentuje delenie bunky C na dve dcérske bunky, S a L.



Obr. 4.3: Upravené delenie vegetatívnych buniek *Anabeana*. Vľavo sekvencia generovaná neparametrickou a vpravo parametrickou L gramatikou.

Tak ako je zobrazené na obrázku 4.3, druhá L gramatika špecifikuje postupnosť vývojových stavov cez ktoré musia bunky prejsť, aby sa dostali do deliace stavu C [23]. Dcérske bunky majú rozličné pozície v modeli a teda čas potrebný na ďalšie rozdelenie bude taktiež iný. Pre dlhé bunky L sú teda potrebné štyri časové jednotky. Pre bunky S je to o jednu časovú jednotku viac.



Obr. 4.4: Klasifikácia gramatík.

4.3 Parametrické L gramatiky

Postupnosť vývojových štádií na obrázku 4.3 výrazne zvyšuje veľkosť abecedy výslednej L gramatiky a taktiež počet pravidiel, ktoré obsahuje množina P. Dochádza tak jednoznačne k zbytočnej komplikácii výslednej d0L gramatiky. *Parametrické L gramatiky* poskytujú riešenie tohto problému pomocou priradenia numerických parametrov k symbolom L gramatík [12].

Definícia 4.6. *Parametrické slovo. Parametrické L gramatiky* pracujú nad *parametrickými slovami. Parametrické slová* sú reťazce *modulov* pozostávajúce zo *symbolov* ku ktorým sú priradené *numerické parametre*. Symboly sú znaky z abecedy V. Parametre sú reálne čísla z množiny $\Re$. Modul so symbolom $A \in V$ a parametrami $a_1, a_2, \ldots, a_n \in \Re$ značíme ako $A(a_1, a_2, \ldots, a_n)$. Každý z modulov patrí do množiny $M = V \times \Re^*$, kde $\Re^*$ je množina všetkých konečných postupností parametrov.

Hodnoty skutočných parametrov korešpondujú k *formálnym* parametrom uvedeným v špecifikácii L gramatiky. Nad parametrami je možné vytvárať *logické* a *aritmetické výrazy*. V nich je možné používať aritmetické operátory $+, -, *, /$; operátor mocniny $\hat{}$, relačné operátory $<, >, =$; logické operátory *not* $!$, *and* $\&$, *or* $|$, a zátvorky $()$. Priorita uvedených operátorov je štandardná. Ďalej sa predpokladá, že logické a relačné výrazy, ktoré sú vyhodnotené ako pravdivé majú hodnotu 1. Naopak, hodnota nepravdivých výrazov je v tomto prípade rovná 0. Znakom $\mathcal{C}(\Sigma)$ označujeme množinu všetkých správne vytvorených logických výrazov s parametrami zo Σ. Množinu všetkých správne vytvorených aritmetických výrazov s parametrami zo Σ označujeme ako $\mathcal{E}(\Sigma)$.

Definícia 4.7. *Parametrická 0L gramatika. Parametrická 0L gramatika* je usporiadná štvorica $G = (V, \Sigma, \omega, P)$, kde

- V je *abeceda* systému,
- Σ je *množina formálnych parametrov*,
- $\omega \in (V \times \Re)^+$ je neprázdne parametrické slovo, ktoré označujeme ako *Axióm*,
- $P \subset (V \times \Sigma^*) \times \mathcal{C}(\Sigma) \times (V \times \mathcal{E}(\Sigma))^*$ je *konečná množina pravidiel*.

Pravidlo takejto gramatiky je rozšírené o podmienku. Tvar pravidla môžeme vyjadriť nasledovne:

$$predecessor : condition \rightarrow successor$$

Predchodca je v tomto prípade jeden modul (symbol L gramatiky ku ktorému sú priradené parametre), *následník* predstavuje postupnosť modulov. *Podmienka* z *angl. Condition* je logický výraz, ktorý určuje kedy môže byť dané pravidlo aplikované na daný modul. Napríklad L gramatika s ktorou sme pracovali aj v predchádzajúcich príkladoch by vyzerala v parametrickom prevedení takto:

```
1 #define LEFT -1
2 #define RIGHT 1
3
4 Axiom: M(5,RIGHT)
5 M(t,p) : t<5 --> M(t+1,p)
6 M(t,p) : t == 5 && p == LEFT --> M(1,LEFT)M(2,RIGHT)
7 M(t,p) : t == 5 && p == RIGHT --> M(2,LEFT)M(1,RIGHT)
```

Ak to všetko zhrnieme pravidlo vybrané pre prepísanie modulu musí spĺňať nasledujúce tri podmienky:

- Symbol modulu a symbol predchodcu vo vybranom module musia byť rovnaké.
- Počet skutočných parametrov modulu je rovnaký ako počet formálnych parametrov predchodcu.
- Podmienka musí byť vyhodnotená ako pravdivá (a to hneď potom ako dôjde k nahradeniu formálnych parametrov skutočnými hodnotami parametrov).

Definícia 4.8. *Derivácia v parametrickej 0L gramatike.* Pokiaľ modul a produkuje parametrické slovo χ ako výsledok prepisovacieho procesu v L gramatike G, potom píšeme $a \rightarrow \chi$. Uvažujme parametrické slovo $\mu = a_1 a_2 \dots a_m$, potom hovoríme, že slovo $v = \chi_1 \chi_2 \dots \chi_m$ je *priamo derivované* z μ, značíme ako $\mu \Rightarrow v$, vtedy a len vtedy ak, $a_i \rightarrow \chi_i$ pre všetky $i = 1, 2, \dots, m$. Parametrické slovo v je generované L gramatikou G deriváciou dĺžky n, pokiaľ existuje postupnosť slov $\mu_0, \mu_1, \dots, \mu_n$ taká, že $\mu_0 \Rightarrow \omega$, $\mu_n = v$ a $\mu_0 \Rightarrow \mu_1 \Rightarrow \dots \Rightarrow \mu_n$.

Pokiaľ množina pravidiel neobsahuje žiadne odpovedajúce pravidlo je modul vrátane parametrov prepísaný sám na seba. V prípade, že sa k jednému symbolu viaže viacero parametrov predpokladá sa, že hodnota prvého parametru udáva stav korytnačky pri interpretácii pravidiel.

4.4 Gramatiky s kontextovou podmienkou

Nasledujúca podkapitola je venovaná gramatikám s kontextovou podmienkou. Pre tento druh gramatík platí, že jednotlivé gramatické pravidlá sú obohatené o podmienky, ktoré podmieňujú možnosť aplikácie daného pravidla. Podmienky rozdeľujeme na dva základné typy a to na *povoľujúce podmienky* z *angl. Permitting conditions* a *zakazujúce podmienky* z *angl. Forbidding conditions*. Pri aplikácii pravidla sa v prepisovanom reťazci pozeráme ešte na symboly uvedené v jednotlivých podmienkach. Symboly uvedené v povoľujúcej podmienke musí aktuálna vetná forma obsahovať. Naopak symboly uvedené v zakazujúcej časti podmienky sa vo vetnej forme vyskytovať nemôžu. Až v prípade, že sú všetky podmienky

splnené môže dôjsť k prepísaniu aktuálnej vetnej formy. Podľa spôsobu aplikácie jednotlivých pravidiel gramatiky môžeme ďalej tieto gramatiky rozdeliť na *sekvenčné* a *paralelné*. Práve uvedeným sekvenčným a paralelným gramatikám je venovaná nasledujúca časť práce v ktorej ich formálne zadefinujeme a na konkrétnych príkladoch rozoberieme spôsob ich použitia. Pre lepšie pochopenie sú uvedené aj sekvečné formy týchto gramatík hoci nie sú zástupcami L gramatík. Z predstaviteľov L gramatík je v tejto kapitole uvedená len ich paralelná verzia. Definície uvedené v tejto časti sú prevzaté z literatúry [22].

4.4.1 Sekvenčné gramatiky s kontextovou podmienkou

Sekvenčné gramatiky s kontextovou podmienkou boli prvýkrát formálne definované v roku 1970 van der Waltom. *Sekvenčná gramatika s kontextovou podmienkou* predstavuje vlastne bezkontextovú gramatiku avšak s tým rozdielom, že aplikácia jednotlivých pravidiel gramatiky je obmedzená na základe uvedených povoľujúcich a zakazujúcich podmienok. V každom derivačnom kroku môže byť na základe použitého prepisovacie pravidla prepísaný práve jeden neterminál. Gramatika teda aplikuje prepisovacie pravidlá sekvenčným spôsobom.

Definícia 4.9. *Gramatika s kontextovou podmienkou.* Gramatika s kontextovou podmienkou je štvorica $G = (V, T, P, S)$, kde

- V je *úplná abeceda*,
- T je *množina terminálov* ($T \subset V$),
- P je konečná *množina pravidiel*,
- $S \in V - T$ je *štartovací neterminál*.

Pravidlá z množiny pravidiel P sú v tvare $(A \to x, Per, For)$, kde $A \in V - T$, $x \in V^*$ a konečné množiny $Per, For \subseteq V^+$. V prípade, že jedna z množín Per alebo For je neprázdna, teda platí, že $Per \neq \emptyset$ alebo $For \neq \emptyset$, nazývame dané pravidlo *podmieneným* z *angl. conditional*. V opačnom prípade ide o *bezkontextové* z *angl. context–free* pravidlo nakoľko neobsahuje množiny Per a For.

Definícia 4.10. *Stupeň gramatiky.* Gramatika G je stupňa (r, s), kde r a s sú prirodzené čísla, ak pre každé $(A \to x, Per, For) \in P$ platí, že $max(Per) \leq r$ a $max(For) \leq s$.

Podobne ako aj u bezkontextových gramatík aj u gramatík s kontextovou podmienkou je potrebné zadefinovať si *priamu deriváciu*. U tohto druhu gramatík pre ňu musia platiť štyri podmienky, ktoré budú uvedené nižšie spolu s jej formálnou definíciou.

Definícia 4.11. *Priama derivácia.* Nech $u, v \in V^*$ a $(A \to x, Per, For) \in P$. Potom hovoríme, že *u priamo derivuje v* vzhľadom na pravidlo $(A \to x, Per, For) \in P$ v gramatike G, značíme ako $u \Rightarrow_G v[(A \to x, Per, For)]$, za predpokladu, že pre nejaké $u_1, u_2 \in V^*$, musia platiť nasledujúce podmienky:

1. $u = u_1 A u_2$,
2. $v = u_1 x u_2$,
3. $Per \subseteq sub(u)$,
4. $For \cap sub(u) = \emptyset$.

Formálny zápis priamej derivácie môžeme takisto zjednodušiť a namiesto zápisu $u \Rightarrow_G v[(A \to x, Per, For)]$ píšeme $u \Rightarrow_G v$. Analogicky ako u bezkontextových gramatík aj v tomto prípade rozširujeme definíciu priamej derivácie $\Rightarrow_G$ na $\Rightarrow_G^k$ (kde $k \geq 0$), $\Rightarrow_G^+$ a $\Rightarrow_G^*$.

Definícia 4.12. *Jazyk generovaný gramatikami s kontextovou podmienkou.* Nech G je gramatika s kontextovou podmienkou. Jazyk generovaný gramatikou G, značíme ako $L(G)$, je definovaný ako:

$$L(G) = \{w \in T^* \mid S \Rightarrow_G^* w\}.$$

Rodinu jazykov generovanú gramatikami s kontextovou podmienkou stupňa (r, s) označujeme $CG(r, s)$ z *angl. Context–conditional grammars.*

4.4.2 Polo–podmienkové gramatiky

Polo–podmienkové gramatiky z *angl. Semi–conditional grammars* sú jednou z modifikácii gramatík s kontextovými podmienkami. Ide teda o gramatiky s kontextovými podmienkami pre ktoré platí, že kardinalita množín s podmienkami je menšia alebo rovná jednej. Zjednodušene povedané je počet prvkov v množine Per resp. For obmedzený a to maximálne na jednu podmienku. Tento druh gramatík je podrobne popísaný v literatúre [22]. V nasledujúcej časti je uvedená základná definícia spolu s názorným príkladom.

Definícia 4.13. *Polo–podmienková gramatika.* Nech $G = (V, T, P, S)$ je gramatika s kontextovými podmienkami. G nazývame *polo–podmienkovou gramatikou*, tzv. *sc–gramatikou*, ak pre každé pravidlo $(A \to x, Per, For) \in P$ platí $|Per| \leq 1$ a $|For| \leq 1$.

V praxi je potom zaužívaný spôsob značenia pravidiel taký, že sa vynechávajú množinové zátvorky $\{\}$ a namiesto označenia prázdnej množiny $\emptyset$ sa používa symbol 0.

Príklad 4.4.1. Uvažujme nasledujúcu polo podmienkovú gramatiku

$$G = (\{S, A, B, A', B', a, b\}, \{a, b\}, P, S)$$

kde

$$\begin{aligned} P = \{&(S \to AB, 0, 0), && (A \to A'A', B, 0),\\ &(B \to bB', 0, 0), && (A' \to A, B', 0),\\ &(B' \to B, 0, 0), && (B \to b, 0, 0),\\ &(A' \to a, 0, 0), && (A \to a, 0, 0)\}. \end{aligned}$$

Z týchto pravidiel je zrejmé, že symbol A môže byť prepísaný na $A'A'$ iba v prípade, ak sa v prepisovanom reťazci vyskytuje znak B. Podobne aj znak A' môžeme prepísať na A iba v prípade, že prepisovaný reťazec obsahuje B. V prípade, že sa v pravidlách nevyskytuje žiadna podmienka, t.j. $|Per| = 0$ a $|For| = 0$, môžeme takéto pravidlo aplikovať kedykoľvek, akonáhle sa v prepisovanom reťazci vyskytuje znak z ľavej strany pravidla. Výsledkom uvedenej gramatiky je jazyk

$$L(G) = \{a^n b^m \mid m \geq 1, 1 \leq n \leq 2^m\},$$

ktorý nie je bezkontextový.

4.4.3 Paralelné gramatiky s kontextovou podmienkou

Paralelné gramatiky opísané v tejto časti sú založené na *ET0L* gramatikách. *ET0L gramatiky* sú významným predstaviteľom paralelných gramatík v modernej počítačovej vede. Viac o týchto gramatikách je popísané v literatúre [22]. Paralelné gramatiky, ktoré budú prezentované v tejto časti sú teda *ET0L gramatiky* rozšírené o *povoľujúce* a *zakazujúce* podmienky. Ide teda o L gramatiky s povoľujícimi podmienkami.

Definícia 4.14. *ET0L gramatika s kontextovou podmienkou. ET0L gramatika s kontextovou podmienkou* je *t+3*–ica, $G = (V, T, P_1, \ldots, P_t, S)$, kde

- V je úplná *abeceda*,
- T, $(T \subset V)$, je množina *terminálov*,
- P_i, $1 \leq i \leq t$, pre nejaké $t \geq 1$, je konečná *množina pravidiel*,
- $S \in V - T$ je *štartovací neterminál*.

Pravidlá sú v tvare $(a \to x, Per, For)$, kde $a \in V$, $x \in V^*$, a $Per, For \subseteq V^+$ sú konečné jazyky. *CET0L* gramatiku, ktorá neobsahuje mazacie pravidlá nazývame *propagujúca* z *angl. propagating*. Gramatika G je *stupňa (r, s)*, kde r a s sú prirodzené čísla, a pre každé $i = 1, \ldots, t$ a $(a \to x, Per, For) \in P_i$, $max(Per) \leq r$ a $max(For) \leq s$. Nech $u, v \in V^*$, $u = a_1 a_2 \ldots a_q$, $v = v_1 v_2 \ldots v_q$, $q = |u|$, $a_j \in V$, $v_j \in V^*$, a $p_1, p_2, \ldots, p_q$ je sekvencia pravidiel $p_j = (a_j \to v_j, Per_j, For_j) \in P_i$ pre všetky $j = 1, \ldots, q$ a nejaké $i \in \{1, \ldots, t\}$. Ak pre každé p_j platí, $Per_j \subseteq sub(u)$ a $For_j \cap sub(u) = \emptyset$, potom u *priamo derivuje* v vzhľadom na $p_1, p_2, \ldots, p_q$ v G, značíme ako

$$u \Rightarrow_G v\,[p_1, p_2, \ldots, p_q].$$

Definícia 4.15. *Jazyk generovaný ET0L gramatikou. Jazyk generovaný ET0L gramatikou* G je definovaný ako

$$L(G) = \{x \in T^* \mid S \Rightarrow_G^* x\}.$$

4.4.4 Využitie L gramatík s kontextovou podmienkou

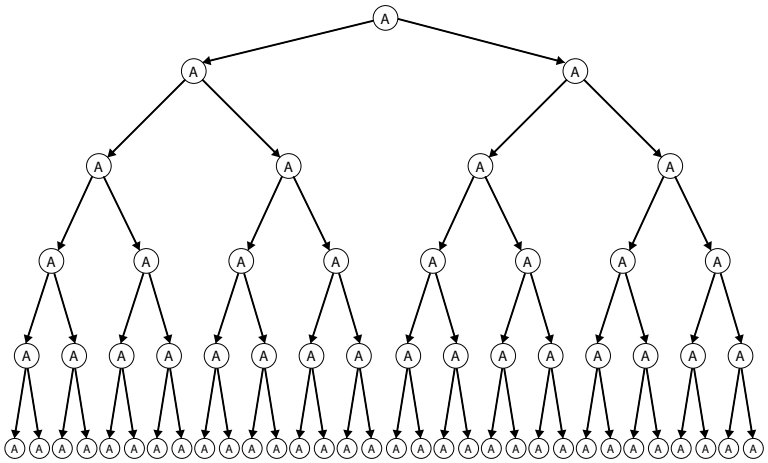
L gramatiky majú veľký potenciál a ich reálne využitie je v mnohých prípadoch bohaté. V nasledujúcej časti sú opísané príklady využitia gramatík s kontextovými podmienkami, ktoré boli podrobne rozobrané v predchádzajúcej časti. Uvedené príklady sa týkajú oblasti mikrobiológie o ktorú je v dnešnej dobe čoraz väčší záujem. Rozobrané sú príklady týkajúce sa vývoja mikroorganizmu, ktorý môže byť ovplyvnený rôznymi vplyvmi ako je nakazenie vírusom, či degeneratívna porucha rastu. Všetky uvedené príklady a ich názorné ukážky sú prevzaté z knihy [22].

Uvažujme bunkový organizmus, ktorého každá bunka sa rozdelí na dve ďalšie bunky, ale iba v prípade, že daný organizmus je zdravý a dochádza tak k jeho zdravému vývoju. V prípade, že sú niektoré bunky tohto organizmu infikované vírusovým ochorením, organizmus stagnuje až pokiaľ nie je vyliečený. Počas stagnácie sa jednotlivé bunky organizmu len reprodukujú a nedochádza tak k produkcii nových buniek. Vývoj takéhoto organizmu môžeme definovať pomocou *polo–podmienkovej L gramatiky*, ktorá bola popísaná v definícii 4.13. Zdravé bunky budeme označovať písmenom A. Bunky nakazené vírusom zasa

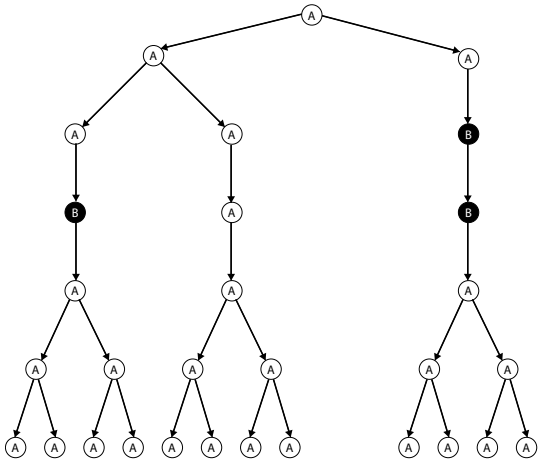
písmenom B. Výsledná gramatika a množina pravidiel P obsahuje nasledujúce pravidlá:

$$(A \to AA, 0, B), \qquad (B \to A, 0, 0),$$
$$(A \to A, B, 0), \qquad (A \to B, 0, 0),$$
$$(B \to B, 0, 0).$$

Obrázok 4.5 a 4.6 ukazuje zdravý vývoj organizmu a následne vývoj organizmu po napadnutí vírusom.



Obr. 4.5: Zdravý vývoj.



Obr. 4.6: Stagnujúci vývoj.

Pomocou L gramatík a vopred definovanej množiny pravidiel sme schopní simulovať rôzne biologické procesy s ktorými sa stretávame pri pozorovaní vývoja mikroorganizmov v čase. Príklad 4.4.2 detailne opisuje štádiá vývoja červených rias z *lat. Red alga*, ktoré patria medzi eukaryotické organizmy schopné fotosyntézy a teda sú zaradené medzi rastliny.

Príklad 4.4.2. Použitím kontextových pravidiel a modifikácie 0L gramatiky môžeme jednoducho popísať zdravý vývoj, degeneratívnu poruchu, či samotnú smrť tohto organizmu. Uvažujme 0L gramatiku $G = (V, P, 1)$, kde $V = 1, 2, 3, 4, 5, 6, 7, 8, [,]$ a množina pravidiel P

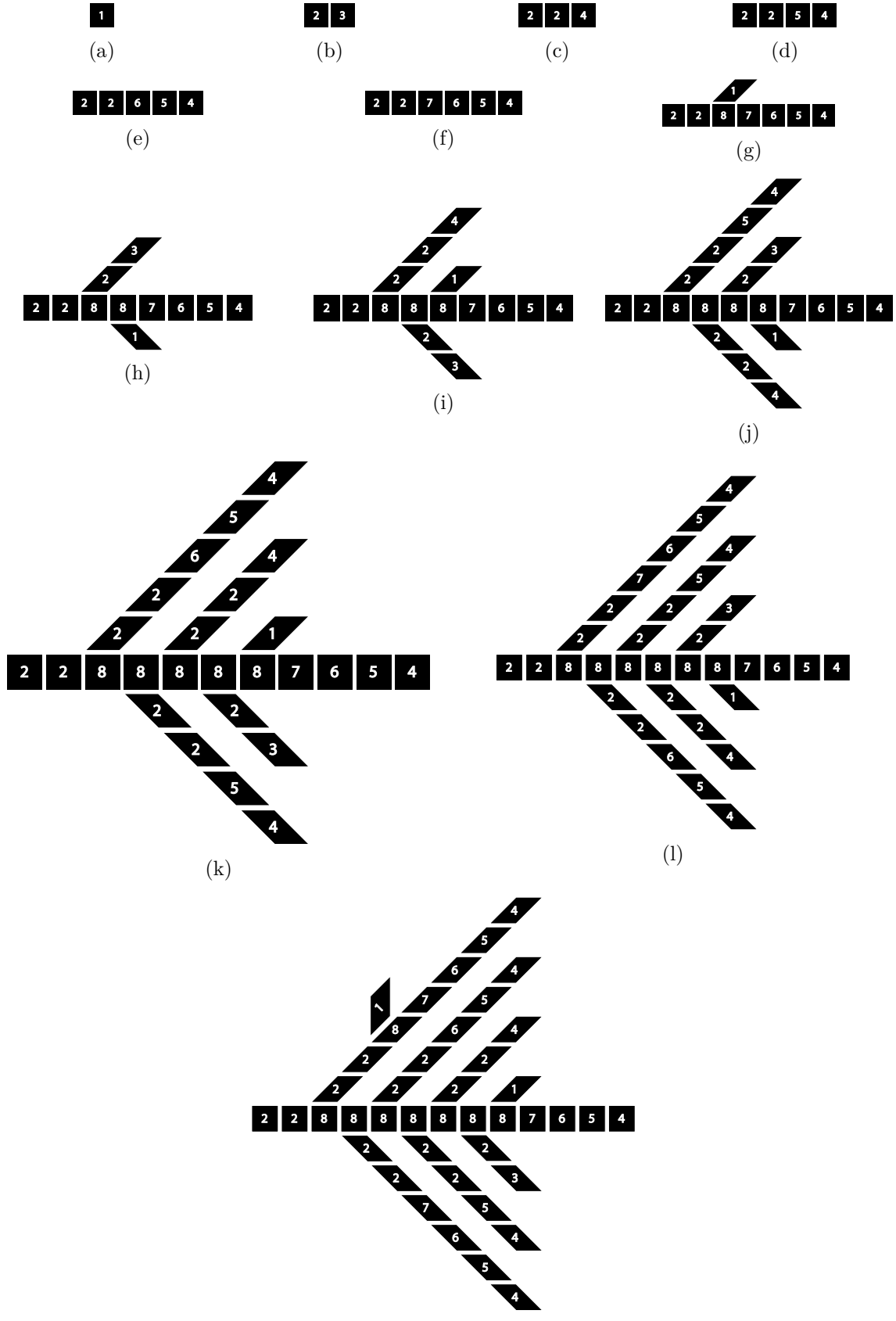
je nasledovná:

$$
\begin{array}{lll}
1 \to 23, & [\to [, & 8 \to 8, \\
2 \to 2, & 5 \to 6, &] \to], \\
3 \to 24, & 6 \to 7, & \\
4 \to 54, & 7 \to 8[1]. &
\end{array}
$$

Z biologického hľadiska, výraz s hranatými zátvorkami 8[1] reprezentuje vetvy, ktorých pozícia je daná znakom 8 v reťazci. Takéto vetvy rastú z vetvy na ktorej vznikli a teda došlo k rozvetveniu. V prípade zdravého vývoja môžeme tento proces zapísať postupnosťou derivácií:

$$
\begin{aligned}
1 &\Rightarrow_G 23 \\
&\Rightarrow_G 224 \\
&\Rightarrow_G 2254 \\
&\Rightarrow_G 22654 \\
&\Rightarrow_G 227654 \\
&\Rightarrow_G 228[1]7654 \\
&\Rightarrow_G 228[23]8[1]7654 \\
&\Rightarrow_G 228[224]8[23]8[1]7654 \\
&\Rightarrow_G 228[2254]8[224]8[23]8[1]7654 \\
&\Rightarrow_G 228[22654]8[2254]8[224]8[23]8[1]7654 \\
&\Rightarrow_G 228[227654]8[22654]8[2254]8[224]8[23]8[1]7654 \\
&\Rightarrow_G 228[228[1]7654]8[227654]8[22654]8[2254]8[224]8[23]8[1]7654.
\end{aligned}
$$

Na základe odvodenej postupnosti derivácií dostávame grafickú reprezentáciu, ktorá je uvedená na obrázku 4.7.



Obr. 4.7: Zdravý vývoj.

Uvažujme, že sa červené riasy, spomínané vyššie, nachádzajú v podmienkach, ktoré neumožňujú zdravý vývin organizmu. Dochádza tak k odumieraniu niektorých častí rastliny. Mladé výhonky nie sú dosť silné na to, aby prežili a tak dochádza k ich odumieraniu. Prežijú len silné staršie vetvy. Výsledná 0L gramatika, obsahuje *zakazujúce* podmienky. Nech $W = a' \mid a \in V$. Potom, $G = (V \cup W, P, 1)$ a množina prepisovacích pravidiel P obsahuje:

$$
\begin{array}{lll}
(1 \to 23, W), & (7 \to 8[1], W), & (3' \to \epsilon, \{4', 5', 6', 7'\}, \\
(2 \to 2, W), & (8 \to 8, W) & (4 \to \epsilon, \emptyset), \\
(3 \to 24, W), & ([\to [, \emptyset), & (5 \to \epsilon, \{4'\}), \\
(4 \to 54, W), & (] \to], \emptyset), & (6 \to \epsilon, \{4', 5'\}), \\
(5 \to 6, W), & (1' \to 2', \{3', 4', 5', 6', 7'\}), & (7 \to \epsilon, \{4', 5', 6'\}), \\
(6 \to 7, W), & (2' \to 2', \emptyset). &
\end{array}
$$

a pre každé $a \in W$

$$
(a \to a', \emptyset), \qquad\qquad (a' \to a', \emptyset).
$$

Ako si môžeme všimnúť vetvy, ktoré sú dostatočne silné na prežitie sú vytvorené symbolmi $2'$ a $8'$. Nakoľko neexistuje pravidlo, ktoré by prepísalo symbol $2'$ na ϵ, tento symbol ostáva zachovaný a nedochádza tak k jeho pomyselnému úhynu. Všetky pravidlá z množiny pravidiel P, ktoré majú na pravej strane znak ϵ reprezentujú mladé výhonky a hynú.

Obdobne sme schopní simulovať aj degeneráciu celej rastliny. Stačí pozmeniť pravidlá a vygenerovaná bude celkom iná rozvetvená štruktúra. Správnou modifikáciou pravidiel sme teda schopný simulovať rôzne vývojové javy rozvetvených štruktúr vyskytujúcich sa v prírode.

V prírode je vývojový proces mnohobunkových organizmov kontrolovaný množstvom látok, ktoré sa vymieňajú medzi jednotlivými modulmi. V prípade rastlín sa jedná o množstvo vody a minerálov, ktoré je absorbované koreňom rastliny. Tieto látky sú následne distribuované z koreňa do vyšších častí. Tento proces sme schopní simulovať rozšírením *parametrických 0L gramatík* o *povoľujúce podmienky* tak, ako je to opísané v časti 4.3. Model vetviacej štruktúry, ktorý využíva takéto prerozdeľovanie zdrojov navrhli Borchert a Honda v [4]. Tento model je regulovaný tokom zdrojov (živín), počínajúc v základe rastliny (koreni). Následne dochádza k šíreniu látok smerom k vrcholom. Vrchol príjme látky a v prípade, že množstvo látok nahromadených vo vrchole dosiahne určitý prah, dôjde k rozvetveniu vrcholu a vytvorí sa tak nová bočná vetva. Distribúcia látok závisí od počtu vrcholov, ktoré vetva obsahuje a typu vetvy. Rastliny obvykle zásobujú hlavné vetvy väčším množstvom látok ako tie bočné. Tento model je jednoduchý a neberie v úvahu počet vrcholov. Rozdeľovanie živín tak nezávisí na veľkosti vetví a bazálny príjem látok je nastavený na konštantnú hodnotu. Preto bude teraz predstavený realistickejší model aj s príslušnou grafickou reprezentáciou. Tento model je založený na kontrolovaní distribúcie látok medzi vetvami vzhľadom na počet vrcholov, ktoré daná vetva obsahuje. Príklad 4.4.3 vychádza z literatúry [22].

Príklad 4.4.3.

$$
\begin{aligned}
axiom: &\quad N(1)I(1, straight, 0, 1)A(1)\\
p_1: &\quad N(k) \rightarrow N(k+1)\\
p_2: &\quad I(id,t,e,c)\,?\,N(k), A(id) : id == 1 \rightarrow I(id, t, \sigma_0 2^{(k-1)\eta^k}, 1)\\
p_3: &\quad I(id,t,e,c)\,?\,N(k), I(id_s,t_s,e_s,c_s), I(id_t,t_t,e_t,c_t) : id == 1 \wedge id_s == 2*id\\
&\quad \wedge id_t == 2*id+1 \rightarrow I(id, t, \sigma_0 2^{(k-1)\eta^k}, c_s + c_t)\\
p_4: &\quad I(id,t,e,c)\,?\,I(id_p,t_p,e_p,c_p), I(id_s,t_s,e_s,c_s), I(id_l,t_l,e_l,c_l) : id_p == [id/2]\\
&\quad \wedge id_s == 2*id \wedge id_l == 2*id+1 \rightarrow I(id, t, \delta(t,e_p,c_p,c), c_s + c_t)\\
p_5: &\quad I(id,t,e,c)\,?\,I(id_p,t_p,e_p,c_p), A(id_a) : id_p == [id/2] \wedge id_a == id\\
&\quad \rightarrow I(id, t, \delta(t,e_p,c_p,c), 1)\\
p_6: &\quad A(id)?I(id_p,t_p,e_p,c_p) : id == id_p \wedge e_p \geq e_t h\\
&\quad \rightarrow [+(\alpha)I(2*id+1, lateral, ep*(1-\lambda), 1)A(2*id+1)]\\
&\quad /(\pi)I(2*id, straight, e_p*\lambda, 1)A(2*id)
\end{aligned}
$$

Táto L gramatika používa nasledujúce typy modulov:

- $I(id,t,e,c)$ je časť stonky s jednoznačným identifikátorom id, kde t je typ stonky, $t \in (straight, lateral)$, e je hodnota živín, c je množstvo vrcholov, ktoré môže daná časť stonky obsahovať,
- $A(id)$ je vrchol ukončujúci časť stonky s id,
- $N(k)$ je pomocný modul, kde k je počet vývojových cyklov, ktoré musia byť uskutočnené do ďalšej derivácie,
- $+(\varphi), /(\varphi)$ je rotácia segmentu o uhol φ (podrobnejšie informácie je možné nájsť v literatúre [22]),
- [a] uzatvárajú sekvenciu bočnej z *angl. lateral* vetvy.

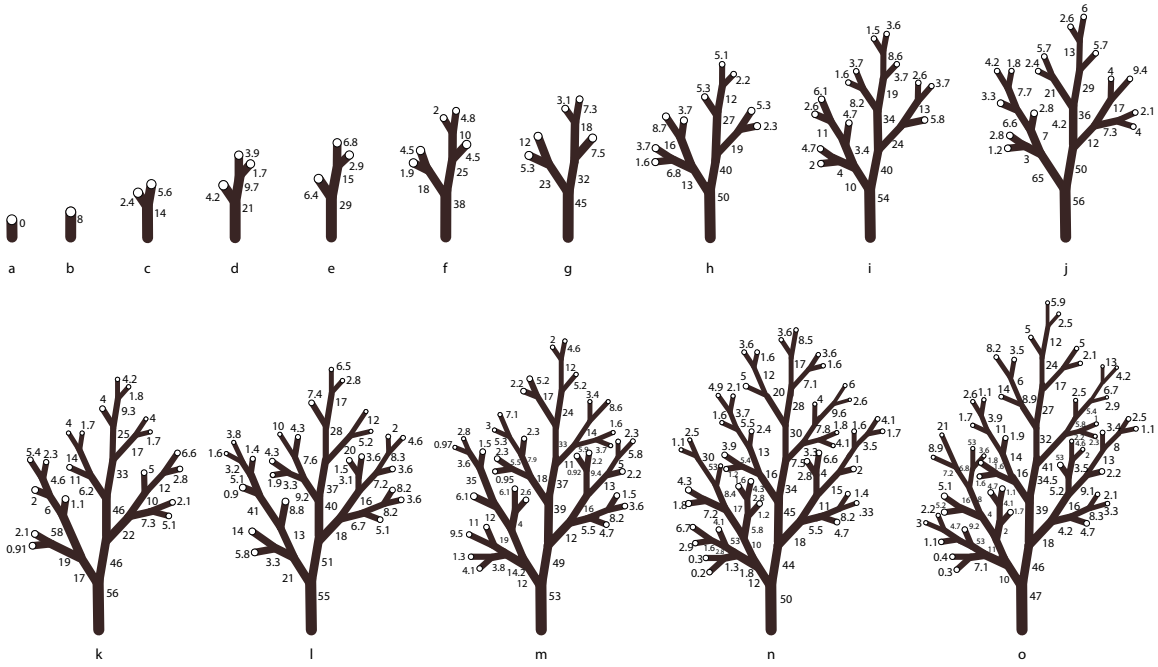
V prípade, že nemôže byť na aktuálny modul použité ani jedno pravidlo, modul sa prepíše sám na seba a tak ostane nezmenený.

Funkcia distribúcie látok δ je definovaná nasledovne:

$$\delta(t, e_p, c_p, c) = \left\{ \begin{array}{ll} e_p - e_p(1 - \lambda((c_p - c)/c)) & \textbf{if t=straight,} \\ e_p(1 - \lambda(c/(c_p - c)) & \textbf{if t=lateral.} \end{array} \right\}$$

Vývin začína z axiómu $N(1)I(1, straight, 0, 1)A(1)$ obsahujúc jednu stonku ukončenú jedným vrcholom. V každej derivácii aplikovaním pravidla p_4, každý vnútorný uzol $I(id, e, t, c)$ obdrží množstvo vrcholov jeho hlavnej vetvy $(I(id_s, t_s, e_s, c_s))$ a bočnej vetvy $(I(id_l, t_l, e_l, c_l))$. Následne je táto hodnota uložená v premennej c. Súčasne dochádza k príjmu živín e_p z rodičovského internódu $I(id_p, t_p, e_p, c_p)$. Distribučná funkcia δ závisí na type vetvy a množstve vrcholov danej vetvy a jej súrodeneckých vetiev. Distribučný faktor λ potom určuje množtvo živín, ktoré budú dopravené do hlavnej vetvy. Množstvo živín, ktoré budú vetve dodané závisí na počte vrcholov. Vzorec $\sigma_0 2^{(k-1)n^k}$ je použitý na simulovanie zvýšenia vstupného toku, kde σ_0 je počiatočné množstvo živín, k je vývojový cyklus a n je konštanta odpovedajúca zmene toku. Pravidlo p_5 má za úlohu prepísať časti stonky, ktoré sú ukončené vrcholmi, pričom zachováva počet vrcholov. Šieste pravidlo p_6 zas kontroluje pridávanie nových častí rastliny. Rozvetvuje časť stonky na dve vetvy, pričom každá z nich je ukončená vrcholom. V tomto modeli si môžeme všimnúť hneď dva toky informácií. Smer *zdola nahor* z *angl. Acropetal flow* má za úlohu distribuovať látky potrebné pre rast rastliny. Opačný

tok smerom *zhora nadol* z *angl. Basipetal flow*, propaguje počet vrcholov, ktoré šíria živiny ďalej. Pozoruhodnou črtou tohto modelu je reakcia rastliny na orezávanie. V prípade, že dôjde k odstráneniu vetvy, model presmeruje živiny do zvyšných vetiev a dôjde k ich rýchlejšiemu rastu. Obrázok 4.8 ilustruje vývojové štádiá modelu uvedeného vyššie.



Obr. 4.8: Vývojové štádiá rastliny generovanej L systémom.

4.5 Obecné využitie L gramatík

L gramatiky predstavujú oblasť teoretickej informatiky, ktorej využitie v oblasti počítačovej grafiky je bohaté. Nakoľko L gramatiky sú predstaviteľmi procedurálneho modelovania, čo znamená, že jednotlivé modely objektov sú popísané pomocou algoritmov, sú vhodné na modelovanie rozsiahlych objektov z reálneho sveta, ktorých 3D skenovanie by bolo príliš zložité a v mnohých prípadoch nereálne. L gramatiky sa používajú na simulovanie rastlín, modelovanie riečnych sietí, ciest a pozemných komunikácií. Taktiež je ich aplikácia vhodná na generovanie modelov budov vo virtuálnom meste. Zaujímavou oblasťou ich využitia je taktiež aplikácia v oblasti počítačového umenia, nakoľko sú schopné generovať lineárne deterministické fraktály [17].

4.5.1 Výhody modelovania pomocou L gramatík

- Hlavnou výhodou L gramatík je, že na základe veľmi malého množstva vstupných dát sú schopné vytvárať relatívne zložité štruktúry. Môžeme teda generovať rozsiahle prírodné scény napr. lesy za použitia iba malého množstva vstupných dát uložených v pamäti počítača.

- L gramatiky umožňujú voľbu počtu iterácií pri aplikovaní prepisovacích pravidiel. To môžeme využiť pri voľbe úrovne detailov generovanej scény. Napríklad pri modeli

mesta, sme schopní meniť úroveň detailov s ohľadom na cieľové požiadavky alebo technické vybavenie.

- Paralelné prepisovanie reťazcov simuluje paralelný vývoj jednotlivých častí konkrétneho objektu. Dochádza tak k súčasnému vývoju všetkých častí napr. organizmu, čo môžeme využiť pri simulácií jeho vývoja v čase.

- Stav objektu v nasledujúcom časovom okamihu $t+1$ získame v jednom kroku pomocou derivácie vety v_t, ktorá popisuje objekt v čase t a následnou interpretáciou vety v_{t+1}. Tento fakt sa využíva pri vytváraní animácií vývoja konkrétneho objektu.

4.5.2 Aplikácia L gramatík

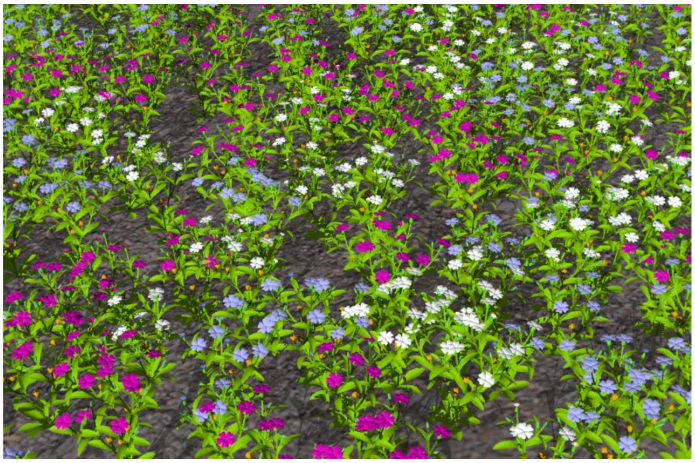
- *Modelovanie rastlín.* Pre modelovanie rastlín boli L gramatiky pôvodne navrhnuté. Práve to bolo hlavným zameraním a jednou z najrozsiahlejších oblastí aplikovania L gramatík.

- *Modelovanie interakcií medzi rastlinami a prostredím.* Pre modelovanie vývoja rastlín prípadne celých ekosystémov je dôležitá nielen ich interakcia s prostredím, ale aj navzájom medzi sebou. Interakcia s prostredím prináša ďalšie možnosti v modelovaní. Predikovať sa dá napríklad úroda alebo predpokladaná ťažba dreva. Vo všeobecnosti rozlišujeme tri základné skupiny interakcií:

  1. Lokálne vlastnosti prostredia – napr. prekážky ovplyvňujúce smer rastu, opora určujúca tvar popínavých rastlín.
  2. Globálne vlastnosti prostredia – dĺžka dňa ovplyvňuje kvitnutie, teplota zas rýchlosť rastu rastliny.
  3. Vzájomná interakcia medzi prostredím a rastlinou – detekcia kolízii, či prístup k svetlu.

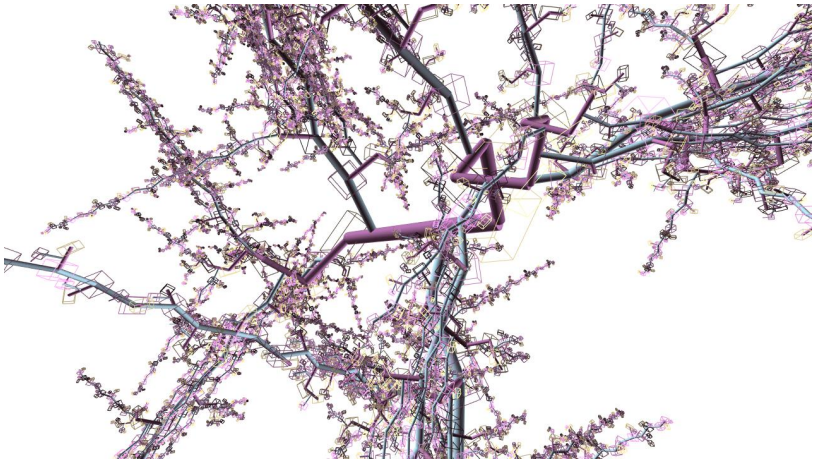
- *Modelovanie ekosystémov.* Ekosystém je prírodné zoskupenie života a životného prostredia [3]. V ekosystéme dochádza k prenosu a obehu hmoty, energie a informácií. Ekosystém predstavuje napr. les, park, záhrada alebo flóra a fauna vyskytujúca sa v lese po lesnom požiari, či ťažbe dreva. Práve L gramatiky poskytujú možnosť simulovať toto dianie v prírode a skúmať tak vplyv prostredia na organizmy, ktoré sa v danom prostredí vyskytujú. Modely týchto ekosystémov sa vyskytujú v mnohých už existujúcich aplikáciách ako aplikácie pre záhradný dizajn, letecké simulátory, hry ale aj aplikácie pre vizualizáciu ekosystémov na vedecké účely. Ukážka ekosystému generovaného pomocou L gramatík je znázornená na obrázku 4.9 [8].

- *Spracovanie prirodzeného jazyka.* Svoje využitie našli L gramatiky aj pri spracovaní prirodzeného jazyka. Prirodzený jazyk je v podstate kontextový jazyk pričom kontext nie je tesný. Pri analýze takéhoto jazyka je potrebné identifikovať vetné členy ako je napr. podmet, predmet, prísudok a ďalšie. Pre tento účel môžeme použiť L gramatiky s kontextovou podmienkou. Uvažujme napr. anglickú vetu „I have a car“, kde I je podmet, have je prísudok a „a car“ je predmet. Člen „a“ a podstatné meno „car“ sú v tesnom kontexte. V prípade, že vetu trochu upravíme „I have a blue car“, dôjde k narušeniu tesného kontextu medzi členom „a“ a podstatným menom „car“. Aj napriek tomu, že neležia bezprostredne pri sebe, patria k sebe a práve to môžeme otestovať pomocou pravidla s konextovou podmienkou.



Obr. 4.9: Ekosystém generovaný L gramatikou.

- *Počítačové umenie.* Schopnosť generovať rôzne fraktály podmienila využitie L gramatík aj v oblasti umenia. Tieto gramatiky sú schopné jednoducho vytvárať obrazce rôznych tvarov, ktoré sú veľmi populárne najmä v abstraktnom umení. Na obrázku 4.10 je ukážka takéhoto obrazu od autora Williama Chyra [6].



Obr. 4.10: 32 L systém.

4.5.3 Modelovanie architektúry

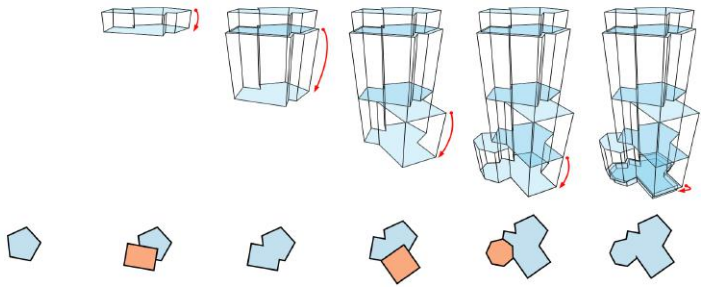
Po stáročia sa architekti inšpirovali tvarmi a geometriou vyskytujúcou sa v prírode. Ich návrhy boli ovplyvnené štruktúrami, proporciami, farbami a vzormi, ktoré sa vyskytujú všade vôkol nás. Všetky tieto fakty následne začlenili do empirického procesu. Práve L gramatiky predstavujú mocný nástroj na modelovanie budov a vytváranie kompletných virtuálnych miest.

Prvou metódou je procedurálne modelovanie miest od Mullera [20]. Na generovanie je použitá stochastická, parametrická L gramatika, ktorá má na starosti generovanie geometrie budov. Jednotlivé primitíva L gramatiky predstavujú geometrické operácie ako zmena merítka, posun a geometrické objekty ako strechy a pod. L gramatiky umožňujú modelovať

rôzne úrovne detailov budov a to už na úrovni topológie objektu, viď. obrázok 4.11. Každá úroveň detailu generovaného objektu odpovedá reťazcu, ktorý je generovaný jedným derivačným krokom v L gramatike. V prípade jednoduchších objektov môžeme úrovne detailov generovať a interpretovať až za behu. V opačnom prípade je možné reťazce reprezentujúce rôzne úrovne detailov predgenerovať a v čase behu aplikácie ich len interpretovať.

Obr. 4.11: Modelovanie budov použitím Mullerovej metódy a L gramatík.

Druhá možnosť je „Procedurálne generovanie pseudo nekonečných miest v reálnom čase“ od Greutera [20]. Budova je budovaná v sekciách. Každá sekcia predstavuje jedinečný plán poschodia, ktoré je potom vytláčané do určitej výšky. Algoritmus začína od strechy, definuje plán poschodia zmenou merítka a rôznymi rotáciami. Následne je plán poschodia vytlačený do určitej výšky. Ďalšia iterácia môže, avšak nemusí, pridať ďalší náhodný tvar k plánu ďalšieho poschodia. Takto sa pokračuje do momentu pokiaľ budova nedosiahne požadovanej výšky. Postupnosť krokov Greuterovej metódy je zobrazená na obrázku 4.12. Na obrázku 4.13 je možné vidieť model takto vygenerovaného mesta.



Obr. 4.12: Modelovanie budov použitím Greuterovej metódy s využitím náhodne extrudovaných tvarov.

Obidve spomínané metódy umožňujú generovať budovy s vysokým stupňom detailov. V kombinácii s dostatočne kvalitnými textúrami sú schopné poskytovať realistické zobrazenia budov.

Obr. 4.13: Mesto generované Greuterovou metódou.

4.6 Zhrnutie

V tejto kapitole boli opísané rôzne druhy L gramatík. Uvedené boli základné definície spolu s názornými príkladmi využitia jednotlivých druhov L gramatík v praxi. Rozdiely medzi prístupmi použitými v bežných bezkontextových gramatikách a L gramatikách sú značné. Nasledujúca časť je preto venovaná porovnaniu ich kľúčových vlastností.

Tak ako bolo spomínane v kapitole 2 bezkontextové gramatiky pracujú sekvenčne. Okrem toho majú bezkontextové gramatiky oproti gramatikám citlivým na kontext dve hlavné nevýhody:

- Nie sú citlivé na kontext.
- Majú menšiu vyjadrovaciu silu než gramatiky citlivé na kontext.

Oproti tomu medzi hlavné výhody bezkontextových gramatík patrí:

- Jednoduchá forma a zápis pravidiel.
- Jednoduché použitie.

Na druhej strane aj gramatiky citlivé na kontext majú svoje klady a zápory. Ich hlavné nevýhody sú:

- Prísne podmienky kladené na okolitý kontext prepísaného symbolu.
- Zložitá forma pravidiel.
- Zložitejšie použitie v praxi.

Za hlavné výhody týchto gramatík sa považuje:

- Ich sekvenčná aj paralelná verzia.
- Reálne využitie v oblasti biológie.

L gramatiky teda predstavujú silný generatívny nástroj a príklady uvedené v tejto kapitole sú toho jasným dôkazom. Pomocou L gramatík s povoľujúcimi podmienkami ako aj parametrických 0L gramatík sme schopní popísať sofistikované modely rastlinných organizmov a to veľmi prirodzeným spôsobom. V porovnaní s gramatikami citlivými na kontext, umožňujú odkazovať na moduly, ktoré nie sú spojené s prepisovaným modulom, čo považujeme za omnoho elegantnejšie riešenie. Tieto poznatky budú ďalej využité pri návrhu a implementácii výsledného systému L gramatík.

Kapitola 5

Návrh systému

5.1 Systém L gramatík

Na formálne zadefinovanie systému L gramatík boli použité poznatky získané z kapitoly 3, ktorá sa zaoberala paralelnými gramatickými systémami. Tieto vedomosti boli skombinované s formalizmami, ktoré sú známe z teórie otvorených L gramatík. Pôjde teda o paralelný gramatický systém zložený z otvorených L gramatík.

Z definície 3.1 plynie, že gramatický systém je zložený z *1* až *n komponent*. Za komponentu sa v tomto prípade považuje *gramatika*, ktorá je súčasťou tohto systému. Počet komponent tvoriacich tento systém bude daný počtom stromov v implementovanej aplikácii. Užívateľ si počet komponent môže navoliť cez užívateľské rozhranie.

Súčasťou definície 3.1 sú taktiež *komunikačné symboly*. Tie slúžia na komunikáciu medzi gramatikami. A teda súčasťou derivácie v takomto systéme sú aj komunikačné kroky. V nami navrhnutom systéme L gramatík sa bude komunikácia vyskytovať vo forme zdieľania informácií medzi prostredím a L gramatikou, teda stromom. *Komunikácia* však nebude prebiehať medzi jednotlivými L gramatikami, ktoré budú súčasťou systému. Komunikačné symboly tak v definícii navrhnutého systému L gramatík nebudú. Komunikačné moduly budú súčasťou definície otvorených L gramatík. Komunikácia bude prebiehať oboma smermi. Životné prostredie bude informovať rastlinu o množstve aktuálne dostupných živín. Rastlina bude informovať prostredie o množstve absorbovaných látok. Po uskutočnení tejto komunikácie dôjde k nastaveniu hodnôt parametrov. Za rast považujeme prepísanie aktuálneho reťazca. Dôležitá v celom systéme bude ukončovacia podmienka. Tá je stanovená počtom derivačných krokov, ktoré sú zadávané užívateľom.

5.1.1 Formálna definícia

Definícia 5.1. *Paralelný systém L gramatík. Paralelný systém L gramatík* Π stupňa $n, n \geq 1$, je $n+2$–tica $\Pi = (N, T, G_1, \ldots, G_n)$, kde

- N abeceda *neterminálnych* symbolov,
- T abeceda *terminálnych* symbolov,
- $G_1 \ldots G_n$ sú zložky systému.

Gramatika G_i je i–tá zložka paralelného gramatického systému Π, pričom ide o 1 až n *otvorených L gramatík*. Paralelný systém L gramatík Π je zložený z postupnosti *parametrických*

bezkontextových L gramatík, def. 4.7, rozšírených o *komunikačné moduly*. Na základe tvrdení uvedených v definícii 3.1, sú jednotlivé zložky tvoriace systém *nezávislé* L gramatiky.

Definícia 5.2. *Konfigurácia sytému. Konfigurácia sytému L gramatík* Π je usporiadaná n–tica $(x_1, x_2, \ldots, x_n)$ s $x_i \in V_\Pi^*$ pre každé $1 \leq i \leq n$, kde $(x_1, x_2, \ldots, x_n)$ sú aktuálne vygenerované reťazce otvorených L gramatík. V_Π je množina zložená z neterminálov a terminálov, teda $V_\Pi \in (N \cup T)$.

Z konfigurácie $(x_1, x_2, \ldots, x_n)$ systém prejde priamo do konfigurácie $(y_1, y_2, \ldots, y_n)$ vtedy, ak platí, že $x_i \Rightarrow y_i$ a v každej komponente G_i, $1 \leq i \leq n$ sú použité prepisovacie pravidlá z množiny prepisovacích pravidiel P_i, okrem prípadu, kedy je x_i terminálny symbol, teda $x_i \in T^*$. Vtedy je modul prepísaný sám na seba, $y_i = x_i$. Množina prepisovacích pravidiel P_i je súčasťou zložkovej gramatiky G_i, teda $P_i \in G_i$.

Definícia 5.3. *Derivačný krok sytému.* Paralelný systém L gramatík $\Pi = (N, T, G_1, \ldots, G_n)$ pre dve n–tice $(x_1, x_2, \ldots, x_n)$ a $(y_1, y_2, \ldots, y_n)$ s x_i a $y_i \in V_\Pi^*$, $1 \leq i \leq n$, kde $x_1 \notin T^*$, derivuje $(x_1, x_2, \ldots, x_n)$ na $(y_1, y_2, \ldots, y_n)$, zapisujeme ako $(x_1, x_2, \ldots, x_n) \Rightarrow (y_1, y_2, \ldots, y_n)$, ak pre každé i, $1 \leq i \leq n$, $|x_i|_K = 0$, a pre každé i, $1 \leq i \leq n$, máme buď $x_i \Rightarrow y_i$ na základe pravidla z P_i, kde $P_i \in G_i$ alebo $x_i = y_i \in T^*$.

5.1.2 Otvorené L gramatiky

Otvorené L gramatiky sú *nedeterministické kontextové parametrické L gramatiky* , ktoré boli navrhnuté pre simuláciu rastu rastlín. Otvorenosť v tomto prípade znamená možnosť interakcie rastliny s prostredím v ktorom sa rastlina vyvíja. Rastlina môže byť ovplyvňovaná globálnymi parametrami prostredia ako je napr. dĺžka dňa prostredníctvom ktorej je kontrolovaný začiatok kvitnutia, denná minimálna resp. maximálna teplota, ktorá ovplyvňuje rýchlosť rastu rastliny a pod. Dochádza teda k prenosu informácií cez telo rastliny, ktorá reaguje na dané podnety z okolia. Na druhej strane rastlina svojimi reakciami ovplyvňuje životné prostredie. Napr. korene pestované v pôde môžu absorbovať vodu v závislosti od koncentrácie vody v ich okolí. To iniciuje tok vody v pôde smerom k vyčerpaným oblastiam, čo ďalej ovplyvňuje rast koreňov. Interakciu medzi rastlinou a prostredím môžeme popísať ako dva súbežné procesy, ktoré navzájom komunikujú a vytvárajú spätnú väzbu. Ide teda o *obojsmernú komunikáciu* medzi rastlinou a životným prostredím. Rastlina obvykle plní nasledujúce funkcie [28]:

- Prijíma informácie o životnom prostredí vo forme skalárov alebo vektorov, tie reprezentujú stimuly, ktoré vnímajú jednotlivé orgány rastliny.
- Prepravuje a spracováva informácie vo vnútri rastliny.
- Generuje odozvy vo forme zmien rastu (napr. rozvoj nových vetiev) a priamy výstup informácií k životnému prostrediu (napr. absorpcia a vylučovanie látok prostredníctvom koreňov).

Na druhej strane je prostredie, ktoré:

- Vníma činnosti rastliny.
- Simuluje interné procesy v prostredí (napr. difúzia látok alebo šírenie svetla).
- Odzrkadľuje zmeny v prostredí v podobe, ktorá je viditeľná rastlinou.

Kvôli prenosu informácií medzi postupnosťou modulov a okolím objektu, ktorý daná L gramatika reprezentuje, boli otvorené L gramatiky rozšírené o tzv. *komunikačné moduly*. Komunikačné moduly sú v tvare:

$$?E(x_1, \ldots, x_m).$$

Formálne môžeme otvorenú L gramatiku zadefinovať nasledovne.

Definícia 5.4. *Otvorená L gramatika. Otvorená L gramatika* G je usporiadaná pätica $G = (V, \Sigma, \omega, P, K)$, kde

- V je *abeceda systému*,
- Σ je množina *formálnych parametrov*,
- $\omega \in (V \times \Re^* \times E)^+$ je neprázdne parametrické slovo, ktoré označujeme ako *axióm*,
- $P \subset (V \times \Sigma^* \times E) \times \mathcal{C}(\Sigma) \times (V \times \mathcal{E}(\Sigma))$ je konečná *množina pravidiel*,
- K je množina *komunikačných modulov*.

Komunikačné moduly sú v tvare $?E(x_1, \ldots, x_m)$, kde $E \in K$ a $x_1, \ldots, x_m \in \Re$. Slúžia na prenášanie informácií medzi L gramatikou G a prostredím. Hodnoty parametrov $x_1, \ldots, x_m$ sú získané na základe komunikácie s prostredím. Postupnosť vykonávaných akcií ilustruje obrázok 5.1, ktorý bol prevzatý z [28].

Definícia 5.5. *Derivačný krok otvorenej L gramatiky. Derivačný krok otvorenej L gramatiky* môžeme rozdeliť do nasledujúcich krokov:

- Prehľadávanie reťazca zľava doprava – hodnoty parametrov nie sú známe, počíta sa nový stav korytnačky v priestore s tým, že nedochádza ku grafickej interpretácii.
- Získanie reálnych hodnôt parametrov – ak sa natrafí na komunikačný modul $?E(x_1, \ldots, x_m)$ potom musí prebehnúť komunikácia s prostredím.
- Nastavenie parametrov – prostredie pridelí parametrom $x_1, \ldots, x_m$ reálne hodnoty, poznáme stav korytnačky.
- Prepisovanie modulov – to je zhodné s *parametrickými L gramatikami* 4.8.

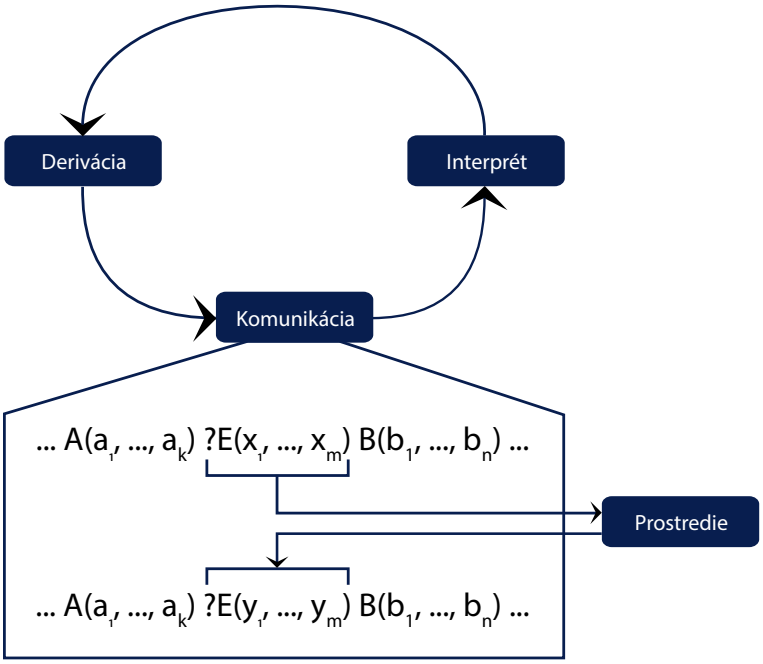
Príklad 5.1.1 opisuje otvorenú L gramatiku, ktorej pravidlá zabraňujú tomu, aby objekt rástol v smere y ďalej, než do vzdialenosti 2 [17].

Príklad 5.1.1.

$$\begin{aligned} \omega &: F(0,0)A?E(0) \\ p_1 &: A \ > ?E(y) \qquad : y < 2 \rightarrow F(x, y+1)A \\ p_2 &: A \ > ?E(y) \qquad : y \geq 2 \rightarrow \epsilon \end{aligned}$$

Modul $F(x, y)$ vygeneruje úsečku z aktuálnej pozície do bodu o súradniciach $[x, y]$. Pravidlo p_1 je aplikované, pokiaľ vrchol A neprekročil hranicu $y = 2$. Pokiaľ k tomu došlo, je aplikované pravidlo p_2, ktoré vrchol A zmaže z postupnosti modulov. Derivácie teda majú nasledujúci tvar:

$$\begin{aligned} F(0,0)A?E(0) &\Rightarrow F(0,0)F(0,1)A?E(1) \Rightarrow \\ &\Rightarrow F(0,0)F(0,1)F(0,2)A?E(2) \Rightarrow \\ &\Rightarrow F(0,0)F(0,1)F(0,2)A?E(3) \end{aligned}$$



Obr. 5.1: Schéma derivácie v otvorenom L systéme.

5.1.3 Využitie

Navrhnutý systém L gramatík môže nájsť svoje uplatnenie v oblasti ekológie. *Ekológia* je vedný odbor, ktorý skúma vzťahy medzi organizmami a životným prostredím, či živými organizmami navzájom [3].

Zavedením otvorených L systémov do paralelného gramatického systému sme schopní simulovať celé ekosystémy. Zložkové L gramatiky môžu popisovať rôzne rastlinné druhy. Pozorované môže byť spolunažívanie rozličných rastlín v prírode. Obojsmerný tok informácií nám umožňuje sledovať nielen vývoj organizmov, ale aj vyčerpanie zdrojov živín v prostredí.

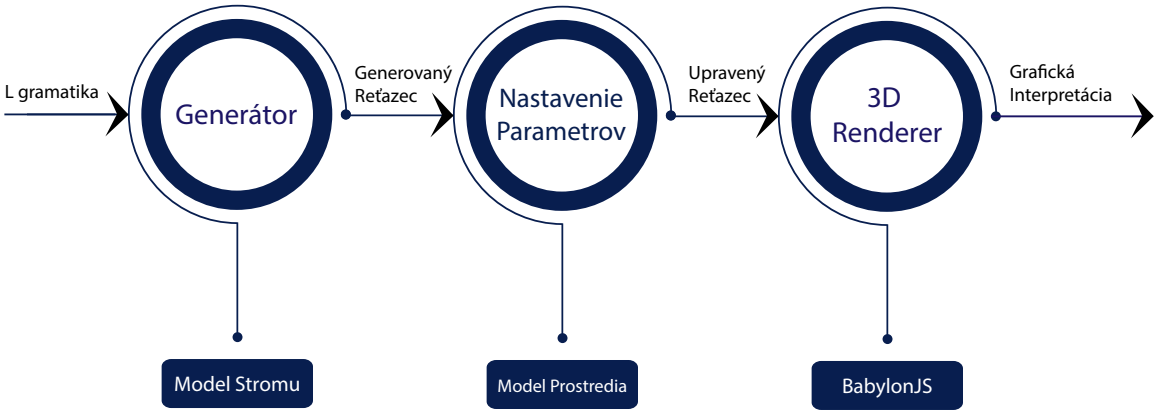
Možné oblasti využitia:

- Modelovanie parkov a lesov.
- Simulovanie vývoja celých ekosystémov.
- Simulovanie vývoja koreňových systémov v závislosti od množstva živín v pôde.
- Simulovanie vývoja korún stromov v závislosti na množstve svetla.
- Detekcia kolízií medzi vetvami.

Navrhnutý systém L gramatík poskytuje formálny základ pre simuláciu zložitejších prírodných dejov. Vďaka komunikácii L gramatík s okolitým prostredím, tak systém môže nájsť uplatnenie nielen v oblasti modelovania, ale hlavne v oblasti simulácie. Zložitosť simulovaných dejov potom závisí na množstve a význame parametrov predávaných medzi prostredím a L gramatikami a takisto na zložitosti modelu, ktorý L gramatika popisuje.

5.2 Návrh implementácie

Implementovaný systém bude zložený z troch hlavných zložiek, tak ako je ilustrované na obrázku 5.2. Prvou je generátor reťazca. Reťazec bude následné interpretovaný a zobrazený v scéne. *Generátor* bude generovať reťazec, ktorý bude popisovať implementovaný model organizmu. V našom prípade pôjde o model stromu, ktorého latinský názov je *Alstonia Scholaris*. Ten bude detailne popísaný v podkapitole 5.3. Z takto vygenerovaného reťazca prostredníctvom grafickej interpretácie získame vizuálnu podobu stromu. Dôjde teda k vykresleniu stromu pomocou 3D objektov na čo bude v našom programe použitá knižnica BabylonJS. Pred samotným vykreslením však bude nutné simulovať odoberanie živín z prostredia. Práve živiny budú mať priamy vplyv na stupeň rozvoja stromovej štruktúry. Systém tak bude obsahovať model prostredia, ktorý bude dodávať živiny vygenerovaným stromom. Použitý model prostredia je bližšie popísaný v časti 5.4. Celý program bude obsahovať užívateľské rozhranie. To bude slúžiť jednak na ovládanie simulácie, ako aj na zadávanie vstupných parametrov. Takisto bude možné pohybovať sa po vykreslenej scéne.



Obr. 5.2: Hlavné časti implementovaného systému.

5.3 Rastlinný model

Modelovanie rastlín je proces, ktorý kombinuje poznatky z biológie, matematiky a počítačovej grafiky na generovanie virtuálnych rastlín. Práve pre túto činnosť boli navrhnuté L gramatiky, ktoré boli podrobne popísané v kapitole 4.

Základom každej L gramatiky je model organizmu alebo rastliny, ktorý ma daná L gramatika popisovať. Obecne platí, že čím komplexnejší model chceme použiť, tým komplikovanejší popis modelu musíme zvoliť. Podrobný popis modelu má potom za následok väčšiu podobnosť a odzrkadľovanie reálneho chovania modelu v simulácii.

Nami navrhnutý systém L gramatík vychádza z poznatkov získaných o strome *Alstonia Scholaris*. Práve tento druh stromu sa stal východiskovým modelom, ktorý bude opísaný v implementačnej časti. Tento model bude najprv rozobratý obecne. Nasledovať bude popis spôsobu odvodenia a získania pravidiel L systému.

5.3.1 Alstonia Scholaris

Alstonia Scholaris je tropický strom svojím pôvodom z Indočíny [36]. Ide o strom s výraznou kôrou. Vo voľnej prírode dosahuje výšky od 18 do 26 metrov. Listy sú oválne, zúbkovité,

zvyčajne o dĺžke až 20 cm. Kôra stromu je hnedo šedá a výrazne horká s tmavými škvrnami. Tento strom je rozšírený najmä v juhovýchodnej Ázii, južnej Číne, na indickom subkontinente a v severnej Austrálii [11]. Kôra tohto stromu sa používala ako písací materiál z čoho je latinský názov. Čaj z lístkov sa používa na liečenie chronických ochorení. Na Srí Lanke sa drevo z tohto stromu používa k výrobe rakiev.

5.3.2 Parametre modelu

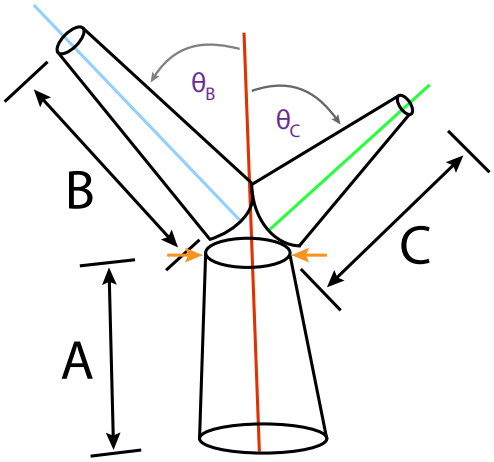
V reálnom svete pozostáva tento strom z viacerých častí a to z kmeňa, vetiev, listov, či kvetov. V modelovanom systéme sa kvôli jednoduchosti sústreďujeme iba na kmeň a ďalšie vetvy. Časti ako listy a kvety zanedbávame z toho dôvodu, že pre účely našej simulácie nie sú kľúčovými prvkami. Naopak parametre, ktoré nás pri simulácii budú zaujímať sú dĺžka vetvy resp. kmeňa a jej hrúbka. Dôležitým faktorom bude pri danom modeli aj uhol vetvenia, ktorý bude mať vplyv na celkový vzhľad stromu. Popis uvedených parametrov spolu s názornými ukážkami boli prevzaté z literatúry [37].

Dĺžka vetvy

Pomer *dĺžky vetiev* z *angl. Length ratio* predstavuje pomer medzi dĺžkou rodičovskej a dcérskej vetvy. V našom prípade sa dĺžka vetiev vypočíta na základe množstva získaných živín z prostredia.

Uhol vetvenia

Uhol vetvenia z *angl. Branching angle* vyjadruje uhol medzi rovinou dcérskych vetiev a rovinou rodičovskej vetvy (obrázok 5.3). Aj v tomto prípade sa prejavujú znaky seba podobnosti rastlín z *angl. Self–similarity pattern*. V simulovanom systéme je možné navoliť si tento uhol. Implementovaný model však počíta s implicitnými hodnotami, ktoré boli odvodené pre tento konkrétny druh stromu. Konkrétne hodnoty týchto parametrov sú uvedené v časti 5.3.3.



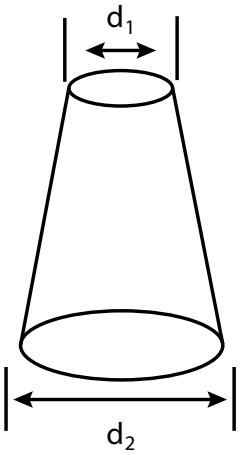
Obr. 5.3: Uhly medzi dcérskymi vetvami B, C a rodičovskou vetvou A.

Hrúbka vetvy

Určuje hrúbku danej vetvy vo vrchnej a spodnej časti, viď. obrázok 5.4. V našom modeli tento parameter interpretujeme iným spôsobom. Hodnota tohto parametru je založená na množstve živín, ktoré daná vetva odoberie z prostredia. Pri vykresľovaní je však hrúbka vetiev počítaná pre celú vetvu ako jeden číselný údaj. Vetva je tak v hornej aj dolnej časti rovnaká kvôli použitiu grafických primitív, ktoré neumožňujú nastavenie oddelenej hrúbky pre vrchnú a spodnú časť. Obecne však pre tento parameter platí vzťah:

$$\frac{d_{child}}{d_{parent}} \tag{5.1}$$

kde d_{child} je hrúbka dcérskej vetvy a d_{parent} zase hrúbka vetvy z ktorej dcérska vetva rastie. Ide teda o jej priameho predka teda rodičovskú vetvu.



Obr. 5.4: Hrúbka rodičovskej a dcérskej vetvy.

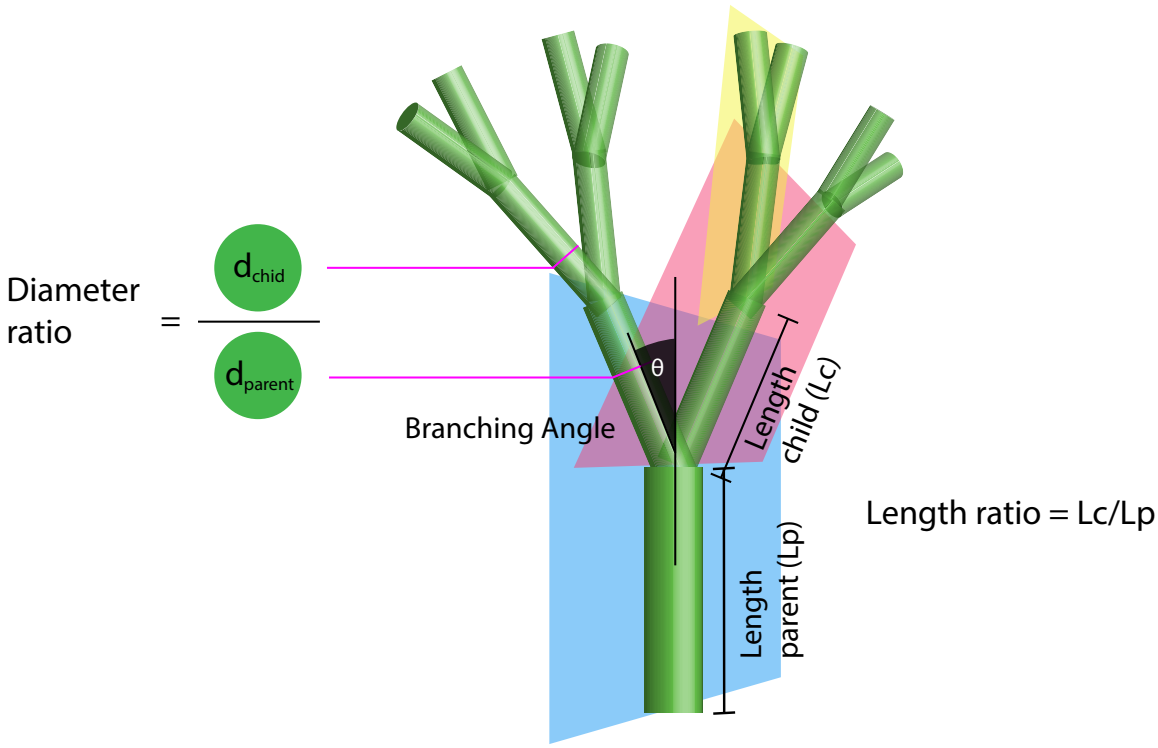
Parametre uvedené v tejto časti sú detailne znázornené na obrázku 5.5. Z uvedeného obrázku plynie, že jednotlivé parametre sa stále počítajú pre určitú časť stromu. Stále teda ide o vzťah medzi dcérskymi vetvami a ich predkom, rodičovskou vetvou z ktorej rastú, čo presne odpovedá princípom vývoja stromu v živej prírode.

5.3.3 Hodnoty parametrov

Parametre pre tento model boli získané z literatúry [37]. Hodnoty parametrov stromu *Alstonia Scholaris* sú uvedené v tabuľke 5.1. Ako si môžeme všimnúť ide o spomínané parametre, ktoré boli podrobne rozobrané v časti vyššie.

Parametre	Hodnota
Pomer dĺžky	8/12
Uhol vetvenia	48°, 40°, 40°, 40°
Pomer hrúbky vetvy	0.95

Tabuľka 5.1: Parametre modelu *Alstonia Scholaris*.



Obr. 5.5: Prehľad parametrov ovplyvňujúcich tvar stromu.

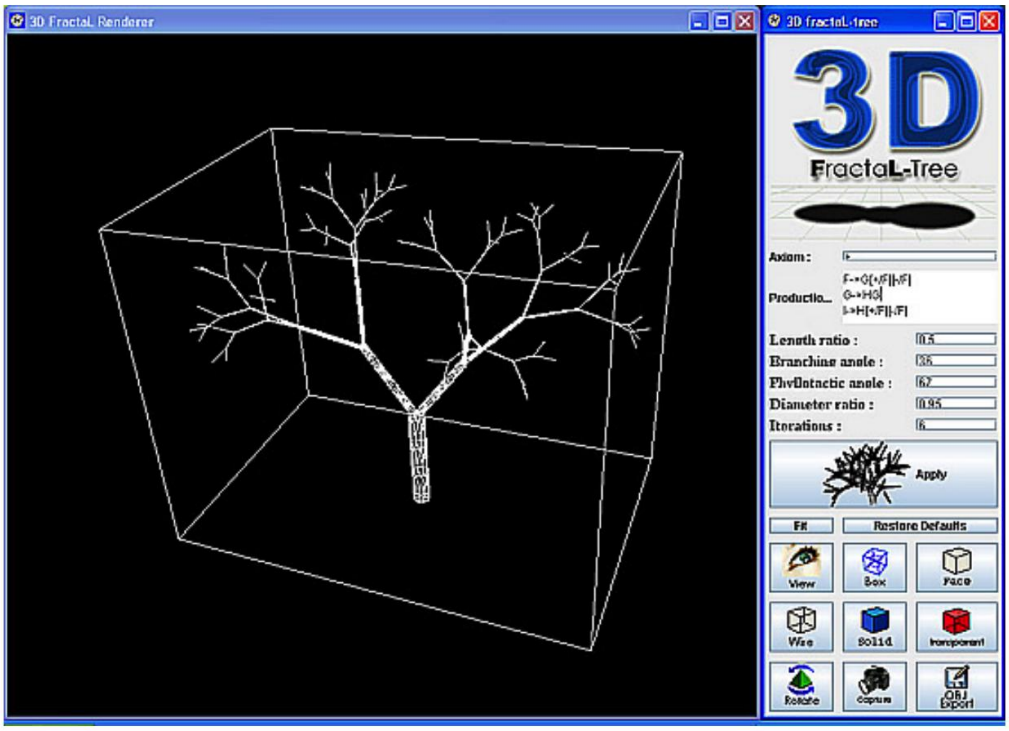
Na základe uvedených parametrov boli vygenerované pravidlá pre L gramatiku, ktorý popisuje tento druh stromu. L gramatika pre *Alstonia Scholaris* vyzerá nasledovne:

$$
\begin{aligned}
&axiom: && F \\
&p_1: && F \rightarrow Y[+++++MF][-----NF][\hat{}\hat{}\hat{}\hat{}\hat{}OF][\&\&\&\&\&PF] \\
&p_2: && M \rightarrow Z - M \\
&p_3: && N \rightarrow Z + N \\
&p_4: && O \rightarrow Z\&O \\
&p_5: && P \rightarrow Z - \hat{}P \\
&p_6: && Y \rightarrow Z - ZY+ \\
&p_7: && Z \rightarrow ZZ
\end{aligned}
$$

V jednoduchosti môžeme povedať, že uvedená L gramatika pozostáva z dvoch druhov pravidiel. O generovanie nových vetiev sa stará pravidlo p_1. Jeho použitie vedie k vytvoreniu štyroch nových vetiev z rodičovskej vetvy. Vetvy sú posunuté o uhol vetvenia. V tomto prípade máme hneď štyri uhly vetvenia v závislosti na smere pohybu korytnačky. Ostatné pravidlá, nepočítajúc počiatočný axióm, majú na starosti úpravu už existujúcich vetiev. V každom kroku tak dochádza k globálnej zmene celého modelu. To má za následok ešte realistickejší vzhľad stromu. Tieto pravidlá však kvôli spôsobu vykresľovania v programe zanedbávame. V programe je strom vykresľovaný po jednotlivých úrovniach. Vykreslené časti sa v priebehu simulácie už nemenia. Na vykresľované časti má v programe hlavný vplyv prostredie a v ňom obsiahnuté živiny. Na základe nich dochádza k prepočítaniu parametrov vetiev pred ich vykreslením v scéne. Parametre stromu sa tak menia v reálnom čase.

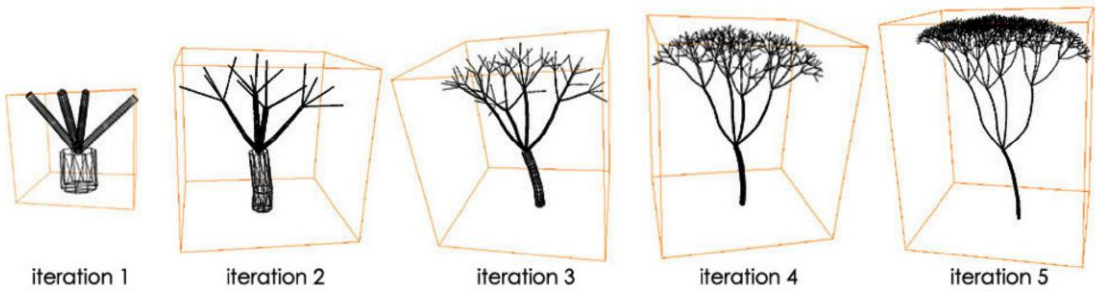
5.3.4 Verifikácia a validácia modelu

Pred samotnou implementáciou navrhnutého systému je vhodné použitý model stromu overiť. *Verifikácia* je proces overovania správnosti modelu vzhľadom k formálnym požiadavkám. V našom prípade ide o popis modelu pomocou zadefinovanej L gramatiky. Druhým procesom je *validácia* modelu. Pri validácii sa snažíme dokázať, že skutočne pracujeme s modelom adekvátnym modelovanému systému [25]. Pre validáciu a verifikáciu uvedeného modelu bol použitý už existujúci softvér viď. obrázok 5.6 [16].

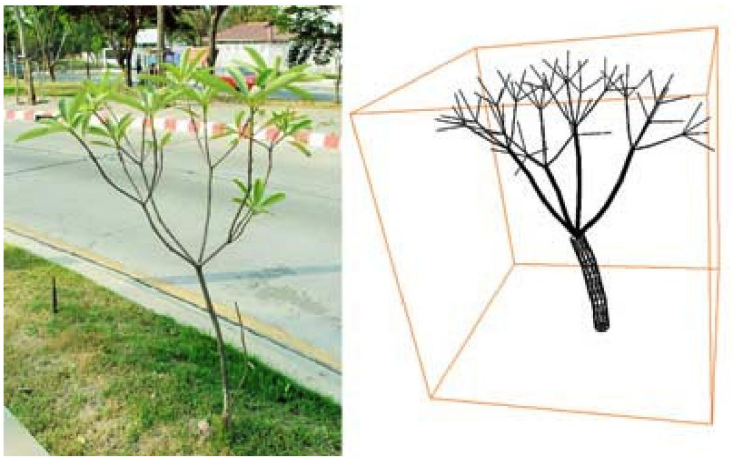


Obr. 5.6: Ukážka výstupu programu 3D Fractal–Tree.

Na obrázku 5.7 môžeme vidieť postupnosť jednotlivých derivácii a vývoj rastlinného modelu za použitia tohto softvéru. Z obrázku 5.8 je už na prvý pohľad badateľné, že vygenerovaný model odpovedá reálnemu stromu. Takto definovaný model stromu môžeme teda považovať za správny. Obrázky 5.6, 5.7 a 5.8 boli prevzaté z literatúry [37].



Obr. 5.7: Postupnosť derivácií pre model stromu *Alstonia Scholaris*.



Obr. 5.8: Porovnanie reálneho stromu (vľavo) a vygenerovaného stromu pomocou L gramatiky (vpravo).

5.4 Model prostredia

Prostredie je druhou zložkou navrhnutého systému L gramatík. Množstvo látok v prostredí, resp. živín, ktoré sú nevyhnutné pre rast a vývin rastlinných organizmov vyskytujúcich sa v danom prostredí, výrazne vplýva na celkový vzhľad stromu resp. celej skupiny stromov.

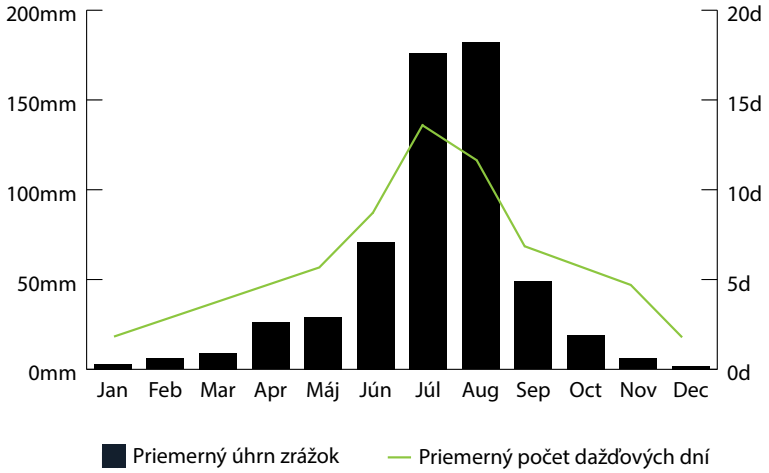
Oblasti v ktorých sa *Alstonia Scholaris* vyskytuje vo voľnej prírode je hneď viacero, zväčša však ide o juhovýchodnú Áziu. Práve na túto oblasť sme sa zamerali v modelovanom systéme. Sústredili sme sa na oblasť Číny, konkrétne na územie okolo hlavného mesta Peking.

Životné prostredie tak predstavuje potrebný zdroj živín. Pre rastlinné organizmy vo všeobecnosti platí, že odoberajú živiny z pôdy. V navrhnutom systéme predstavuje živiny voda. Tá je v priebehu simulácie postupne doplňovaná do prostredia. Množstvo vody, ktorú je potrebné doplniť, je nutné dopočítať. Výpočty boli uskutočňované na základe reálnych údajov získaných z dostupných meteorologických meraní pre túto oblasť [13]. Na základe údajov priemerných zrážok pre konkrétny kalendárny mesiac bol vypočítaný priemerný úhrn zrážok v danom ročnom období. Na základe zvoleného ročného obdobia tak dochádza ku generovaniu a následnému dopĺňaniu živín do prostredia.

Obrázok 5.9 znázorňuje priemerný mesačný úhrn zrážok pre oblasť Pekingu spolu s priemerným počtom dažďových dní. Použité hodnoty boli získané spriemerovaním údajov jednotlivých mesiacov. Na základe toho bol odvodený priemerný úhrn zrážok v danom ročnom období. Vypočítané hodnoty obsahuje tabuľka 5.2.

Ročné obdobie	Zahrnuté mesiace	Priemerný úhrn zrážok (mm)
Jar	Apríl, Máj	27,5
Leto	Jún - August	143
Jeseň	September, Október	34
Zima	November - Marec	4,3

Tabuľka 5.2: Vypočítaný priemerný úhrn zrážok v danom ročnom období.



Obr. 5.9: Priemerný úhrn zrážok pre oblasť Pekingu.

Z vypočítaných hodnôt je zrejmé, že stromy budú mať najväčší prísun živín v lete. Vtedy dôjde k výraznému vývinu. Naopak najslabším obdobím pre rast bude zima. Stromy budú svoj tvar meniť len minimálne. Jar a jeseň sú svojimi hodnotami podobné. Očakávať môžeme podobný priebeh vývoja v spomínaných ročných obdobiach. Aktuálne množstvo živín v prostredí tak priamo vplýva nielen na stupeň vývoja, ale aj na počet vygenerovaných stromov.

Kapitola 6

Implementácia

Nasledujúce podkapitoly detailne rozoberajú spôsob implementácie navrhnutého systému L gramatík. Podrobne je popísaný spôsob implementácie vnútornej logiky programu. Uvedené sú zaujímavé časti implementácie. Posledná časť je venovaná grafickému výstupu a užívateľskému rozhraniu, ktoré je k dispozícií na ovládanie simulácie.

6.1 Použité technológie

Pri výbere použitých technológií bol kladený dôraz na čo najlepšiu prenositeľnosť programu. Ako implementačný jazyk bol zvolený *JavaScript* [14]. Kvôli lepšej testovateľnosti a ladeniu chýb bola použitá nadstavba *Flow* [9]. Práve *Flow* nám umožňuje statickú kontrolu typov v JavaScripte, čo urýchľuje odstraňovanie chýb a vedie tak k rýchlejšiemu procesu implementácie.

6.1.1 React

React [10] je knižnica v JavaScripte pre tvorbu užívateľských rozhraní. Táto knižnica nám umožňuje rozdeliť aplikáciu, resp. užívateľské rozhranie implementovanej aplikácie, do oddelených komponent. Každá komponenta si udržuje svoj vnútorný stav. Užívateľ svojou interakciou s prvkami užívateľského rozhrania mení stav komponent a dochádza tak k prekresľovaniu prvkov GUI. Tento reaktívny prístup bol využitý pri implementácii spomínanej webovej aplikácie. Okrem toho, že komponenty sú znovu použiteľné, môžeme ich ľubovoľne vnorovať a vytvárať tak komplexné užívateľské rozhrania. Výhodou tejto knižnice je veľká podpora ďalších rozšírení a balíčkov. Tie môžu byť jednoducho doinštalované pomocou správcu balíčkov *npm*.

V aplikácii je použitá knižnica *lodash* [7], pre lepšiu manipuláciu so zložitejšími štruktúrami ako sú vnorené objekty, či viacdimenzionálne polia. Ovládacie prvky aplikácie sú vytvorené pomocou knižnice *Material–UI*.

6.1.2 BabylonJS

Na grafickú vizualizáciu bola použitá knižnica *BabylonJS* [5]. Ide o JavaScript framework pre prácu s 3D grafikou. Celý framework je postavený na základoch *WebGL* [34]. Medzi jeho hlavné výhody patrí transparentná podpora *WebGL 1.0* a *WebGL 2.0*.

Scéna je reprezentovaná ako graf, ktorý obsahuje svetlá, kamery, materiály animácie, zvuk, či akcie. Výhodou frameworku je jednoduchá práca s grafickými objektami a optima-

lizované matematické operácie nad vektormi a maticami. Práve rýchle násobenie matíc je nevyhnutné pre dostatočne rýchlu grafickú interpretáciu reťazca.

Framework ďalej obsahuje pokročilé funkcie ako je fyzika alebo antialiasing. V neposlednom rade je súčasťou frameworku aj hardvérová akcelerácia na GPU, ako aj pokročilé optimalizácie vykresľovanej scény.

V našom programe sme využili najmä prepracovaný systém kamery, textúr a grafické primitíva na vykresľovanie stromov. Taktiež bola použitá hmla na zmenšenie záťaže pri vykresľovaní objektov, ktoré sú príliš vzdialené od aktuálnej pozície kamery.

6.2 Vnútorná logika

Nasledujúce podkapitoly detailne opisujú spôsob implementácie vnútornej logiky programu. Opísaný je spôsob generovania času ako aj samotného reťazca, udržiavanie vnútorného stavu aplikácie a použité dátové štruktúry pre uloženie reťazca stromu. Ďalej je v tejto podkapitole rozobraný spôsob grafickej interpretácie vygenerovaného reťazca. V časti 6.3 je spomenuté užívateľské rozhranie programu, ktoré je detailnejšie popísané v prílohe B aj s grafickou ukážkou.

6.2.1 Vstupné parametre

Pred spustením aplikácie je možné nastaviť počiatočné parametre simulácie. Architektúra aplikácii, vytvorených pomocou knižnice *React*, je založená na uchovávaní si vnútorného stavu. Stav je v tomto prípade objekt obsahujúci dvojice *kľúč–hodnota*. Stav má v sebe obvykle uložená najvrchnejšia komponenta. Najvrchnejšou komponentov nášho programu je trieda `App`. Tá má za úlohu spravovať stav celej aplikácie. Táto trieda má v svojom stave uložené vstupné parametre simulácie. Okrem toho stav aplikácie obsahuje aktuálny simulačný čas. Prehľad dostupných parametrov spolu s rozsahom hodnôt a stručným popisom je uvedený v tabuľke 6.1.

Parameter	Rozsah hodnôt	Popis
Derivations	1-100	Počet derivačných krokov
Split Angle	1-90°	Uhol vetvenia
Nutrients	1-1000	Počiatočné množstvo živín
Tree Count	1-100	Počet vygenerovaných stromov
Weather Season	Spring, Summer, Autumn, Winter	Simulované ročné obdobie

Tabuľka 6.1: Vstupné parametre programu.

Treba podotknúť, že nastavené parametre sa zohľadnia iba v prípade, že je simulácia spustená od začiatku. V prípade, že došlo k pozastaveniu simulácie je možné meniť iba aktuálny počet vygenerovaných derivácií. Zmena ostatných parametrov sa v už spustenej simulácii neprejaví, nakoľko by to ovplyvnilo všetky vykreslené reťazce. Muselo by dôjsť ku kompletnej grafickej reinterpretácii vygenerovaných stromov.

6.2.2 Generovanie času

Generovanie času je nevyhnutné pre priebeh simulácie a následný vývoj organizmu. Generátor času reprezentuje premenná `this.timer`. Tá je súčasťou triedy `App`. K inicializácii premennej `this.timer` dochádza v momente spustenia simulácie. Stlačením tlačidla RUN dôjde k volaniu metódy `setInterval()`. Každú sekundu je inkrementovaná hodnota aktuálneho času. Ten je uložený v stave aplikácie, pod kľúčom `actualTime`. K takémuto zvyšovania času dochádza až do momentu, kým sa nevygeneruje požadovaný počet derivačných krokov.

Po dobehnutí simulácie je možné krokovať simuláciu manuálne. V takomto prípade dochádza k zmene simulačného času na základe akcie od užívateľa. Ten mení hodnotu premennej `actualTime`. Pri krokovaní je podporovaná funkcia kroku späť.

6.2.3 Trieda SimulationState

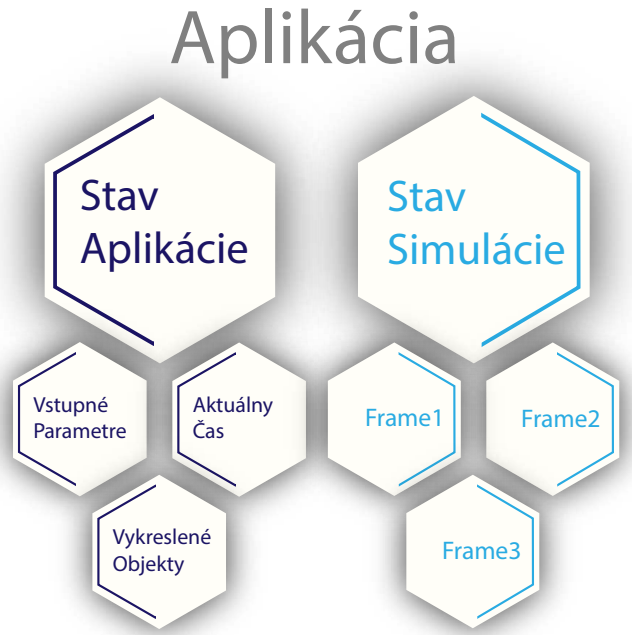
Jadrom celého programu je trieda `SimulationApp`. V nej sa vytvára inštancia triedy `SimulationState` s navolenými parametrami. Trieda `SimulationState` reprezentuje stav simulácie. Ide o vnútorný stav simulácie, ktorý je odlišný od vnútorného stavu aplikácie viď. obrázok 6.1. V simulačnom stave sa ukladajú postupnosti snímkov, ktoré sú najprv spočítané a následne vykreslené v aplikácii. Jednotlivé snímky sa ukladajú do poľa `frames`.

Pole je tvorené položkou `time`, tá predstavuje konkrétny čas a položkou `state`, ktorá obsahuje už vygenerované informácie na základe počtu a typu stromu. V poli `frames` máme pre daný čas stále k dispozícii konkrétny stav simulácie. Známe je tak množstvo a štruktúra vygenerovaných stromov. To nám uľahčí prácu neskôr, pri samotnom vykresľovaní vygenerovaných reťazcov v scéne. Nakoľko ide o manipuláciu s poľom objektov, trieda implementuje požadované metódy na pridávanie nových snímkov do poľa `putFrame()`, či získanie konkrétneho snímku na základe zadaného času `getFrameByTime()`. Vďaka tejto štruktúre uloženia dát nám v simulácii stačí pracovať s aktuálnym časom. Na základe neho už vieme získať všetky potrebné informácie pre vykreslenie objektov v scéne.

Počítanie snímkov je spustené metódou `generate()`. Na základe zvoleného počtu derivačných krokov sa určí počet snímkov, ktoré musia byť vygenerované. Následne je volaná metóda `calculateNextFrameOfApp()`, ktorá spočíta stav (snímok) pre ďalšiu deriváciu. Nevýhodou je, že pred spočítaním nového stavu sa vytvára hlboká kópia aktuálneho snímku. Nakoľko ide objekt s vnorenými objektami, je náročnosť tejto operácie citeľná najmä v deriváciách s vyšším poradovým číslom. Vytváranie kópie objektu bolo urýchlené použitím optimalizovanej verzie funkcie `cloneDeep()` z knižnice lodash.

6.2.4 Trieda AbstractTree

Kvôli jednoduchej rozšíriteľnosti programu o ďalšie modely stromov boli implementované triedy `AbstractTree` a `AbstractTreeNode`. Predpokladá sa, že každý doimplementovaný model stromu bude odvodený dedením práve z triedy `AbstractTree`. Je potrebné implementovať metódu `getLeaves()`, ktorá má ako jediný parameter inštanciu triedy `AbstractTreeNode`. Inštancia tejto triedy reprezentuje abstraktný uzol stromu. Každý uzol stromu musí mať špecifikované svoje súradnice v osi x, y a z. Okrem toho si uzol v sebe ukladá informáciu o svojom priamom predkovi. Táto dvojica údajov nám postačuje na to, aby sme mohli úspešne vykresliť celú štruktúru stromu v 3D priestore.



Obr. 6.1: Rozdiel medzi stavom aplikácie a simulácie.

6.2.5 Generovanie živín

Model prostredia bol implementovaný pomocou triedy `ContextState`. Trieda obdrží informácie o navolených parametroch. Konkrétne ide o hodnotu počiatočného množstva živín a zvolené ročné obdobie. Trieda ďalej obsahuje metódu `generateFlux()`. Tá slúži na manipuláciu s aktuálnym množstvom živín v prostredí, ktoré je uchované v premennej `this.flux`. Metóda `generateFlux()` sa stará o doplňanie nových živín do prostredia. Množstvo pridaných živín sa počíta podľa vzťahu:

$$Random(PRUZ) * treeCount \tag{6.1}$$

kde $PRUZ$ je priemerný ročný úhrn zrážok pre zvolené ročné obdobie a $treeCount$ je počet stromov, ktoré sa majú vygenerovať. Experimentovaním bolo zistené, že násobenie množstva živín počtom požadovaných stromov vedie ku komplexnejšiemu vývoju celého ekosystému. Okrem generovania táto metóda rovno aktualizuje aktuálnu hodnotu živín v prostredí. Prostredie tak ovplyvňuje parametre modelu, ktoré sa menia v reálnom čase.

6.2.6 Generovanie reťazca

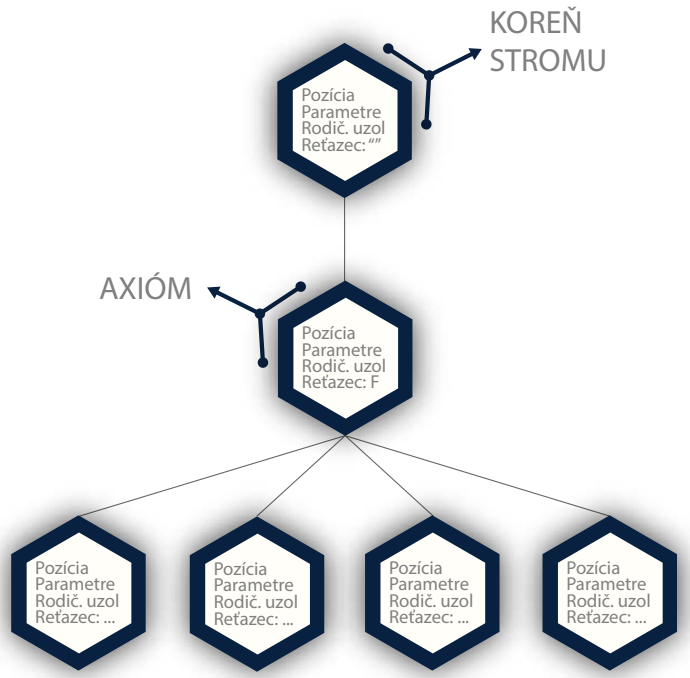
Vygenerovaný reťazec odpovedá popisu modelu stromu. Práve preto je generovanie reťazca implementované v triede `TreeAlstoniaScholaris`. Reťazec sa skladá zo symbolov používaných pri interpretácii korytnačou grafikou. Konkrétne ide o symboly F, $+$, -, $\hat{}$ a $\&$. Ich význam je detailne opísaný v knihe [27].

Generovanie začína z počiatočného axiómu, nasleduje výber a použitie zvoleného pravidla. Jednotlivé pravidlá má v sebe uchovaný implementovaný model. Generovaný reťazec je uložený v stromovej štruktúre. Pri aplikácii pravidla so symbolom F na ľavej strane, dochádza k vytvoreniu nových uzlov n–nárneho stromu[1]. Uzly v sebe uchovávajú informácie

[1] n=4 pre implementovaný model stromu

o svojom predkovi, aktuálny reťazec, súradnice a uhly natočenia. Aktuálny reťazec uzlu tvorí nahradená pravá strana aplikovaného pravidla.

Výsledná dátová štruktúra je uvedená na obrázku 6.2. Na obrázku je možné vidieť, že každý uzol má štyroch nových potomkov, kde každý z potomkov si v sebe uchováva svoju časť reťazca. Tento fakt neplatí pre koreňový uzol, ktorý má len jedného potomka a reprezentuje kmeň. Takéto generovanie reťazca nám umožňuje vykresľovanie stromu po jednotlivých úrovniach. Jedna úroveň potom odpovedá jednému derivačnému kroku. V pravidlách L gramatiky, ktorý popisuje model stromu sa vyskytujú aj ďalšie symboly ako *M*, *N*, *O* a *P*. Tie v implementácii neberieme v úvahu nakoľko by sa musel prepočítať celý vygenerovaný strom. Zohľadnenie týchto pravidiel by mohlo byť jednou z možností ďalšieho vývoja projektu.



Obr. 6.2: Stromová reprezentácia vygenerovaného reťazca.

6.2.7 Interpretácia reťazca

Stromová reprezentácia vygenerovaného reťazca nám uľahčuje proces vykresľovania. Uzly stromu v sebe uchovávajú súradnice. Ide teda o body v 3D priestore. Vďaka uchovaniu si predka vieme potom spočítať smerový vektor danej vetvy. Na základe uložených uhlov dôjde k natočeniu vektora správnym smerom. Použité sú rotačné matice z literatúry [27]. Matice boli rozšírené o homogénne súradnice. Poslednou časťou pred samotným vykreslením vetvy je nastavenie dĺžky a hrúbky vetvy. Tieto parametre sa nastavujú na základe dostupných živín z prostredia.

6.3 Užívateľské rozhranie

Aplikácia obsahuje jednoduché užívateľské rozhranie. To je zložené z dvoch častí. Prvou časťou je menu umiestnené v pravej hornej časti obrazovky. Menu obsahuje parametre simulácie. Okrem parametrov sa v časti SIMULATION nachádzajú ovládacie prvky.

Na ovládanie programu má užívateľ k dispozícii tlačidlá. Celá simulácia sa spúšťa tlačidlom RUN. Počas behu simulácie môže užívateľ kedykoľvek simuláciu prerušiť. V tomto režime je možné krokovať priebeh vývoja stromov. Pomocou tlačidiel NEXT resp. PREVIOUS je možné vykresliť nasledujúci resp. predchádzajúci stav simulácie. To je docielené jednoduchým inkrementovaním resp. dekrementovaním počítadla derivačných krokov. Na znovu spustenie simulácie je k dispozícii tlačidlo RESTART. To je zobrazené aj v prípade, že simulácia dobehne dokonca bez prerušenia užívateľom.

Okrem toho sú v menu zobrazované dodatočné informácie. Ide o aktuálny simulačný čas, aktuálne množstvo živín a množstvo vygenerovaných živín, ktoré boli v danom kroku pridané do prostredia.

Druhou časťou je scéna do ktorej sú vykresľované vygenerované stromy. Tá obsahuje zdroj svetla, kameru, zem a oblohu. Scéna je obohatená o hmlu, čo zlepšuje výkonnosť pri vykresľovaní vzdialených objektov. Kameru je možné ovládať dvoma spôsobmi. Prvým je klasické ovládanie myšou. Druhou možnosťou je využitie šípok. Po scéne sa dá pohybovať aj počas prebiehajúcej simulácie. Takýto pohyb však kvôli výpočtom na pozadí a súčasnému vykresľovaniu grafických primitív nemusí byť plynulý. Ukážka užívateľského rozhrania aplikácie je spolu s jeho detailným popisom uvedená v prílohe B.

Kapitola 7

Testovanie

Táto kapitola obsahuje výsledky získané testovaním implementovanej aplikácie. Proces testovania bol rozdelený na dve fázy. Prvou bolo testovanie aplikácie z hľadiska simulácie. Sledovaný bol vývoj implementovaného modelu stromu vo zvolenom ročnom období. Ďalej bol porovnaný čas potrebný na generovanie ďalšieho derivačného kroku celého systému s časom potrebným na grafickú interpretáciu vygenerovaných reťazcov.

7.1 Testovacie prostredie

Testovanie prebiehalo na vlastnom notebooku s operačným systémom macOS Sierra High, procesorom Intel Core i5 2.7 GHz a grafickou kartou Intel Iris Graphics 6100. Testovaná aplikácia bola spúšťaná vo webovom prehliadači Google Chrome, verzia 65 s parametrami uvedenými v tabuľke 7.1. Namerané údaje sa tak môžu v závislosti od použitého hardvéru a prehliadača líšiť.

Parametre	Hodnota
Nutrients	100
SplitAngle	45°
Derivations	10
Trees	30

Tabuľka 7.1: Testovacie parametre programu.

7.2 Testovanie simulácie

Cieľom uskutočnených testov bolo sledovanie vplyvu prostredia na vygenerované stromy. Presnejšie išlo o vplyv živín z prostredia na stupeň vývinu stromov. Testy boli prevádzané pre jednotlivé ročné obdobia s tým, že počet vygenerovaných stromov aj počiatočné množstvo živín v prostredí bol zachovaný. Dosiahnuté výsledky ilustruje graf 7.1. Z grafu vyplýva, že množstvo vykreslených grafických objektov je v zimnom období minimálne. To odpovedá z botanického hľadiska režimu, kedy sú stromy v spánku. Naopak v lete počet vygenerovaných grafických primitív výrazne narastá. Vývoj stromov ovplyvňuje počet živín, ktorých je v lete dostatok. Dochádza tak k výraznému rozvinutiu celého ekosystému. Za grafický ob-

jekt sa v uvedených testoch považuje kmeň alebo časť vetvy. Na grafe je taktiež badateľné, že od ôsmej derivácie dochádza k výraznému poklesu vygenerovaných nových častí stromov. K tomuto javu dochádza kvôli postupnému vyčerpaniu zdroja živín. Prostredie neobsahuje dostatok živín na rozvoj všetkých stromov. Rastú tak len náhodne vybrané stromy. Vývoj stromov v jari a v jeseni je zhruba rovnaký. V spomenutých ročných obdobiach sa množstvo generovaných živín veľmi neodlišuje, čo odpovedá meteorologickým meraniam z ktorých sme pri implementácii vychádzali. Odpovedajúce grafické výstupy z programu sú uvedené v prílohe C.

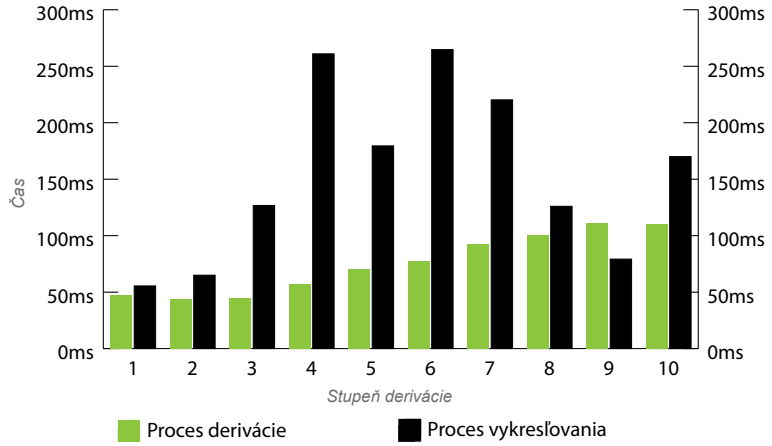
Obr. 7.1: Počet vygenerovaných grafických objektov pri rôznych ročných obdobiach.

7.3 Testovanie časovej náročnosti

Nasledujúca časť sa zaoberá časovou náročnosťou generovania ďalšieho derivačného kroku. Táto operácia v sebe zahŕňa proces komunikácie s prostredím, kedy dôjde k získaniu a nastaveniu skutočných hodnôt parametrov. Namerané údaje sa vzťahujú k derivačnému kroku navrhnutého systému L gramatík. Nejde tak o deriváciu v rámci jednej zložkovej L gramatiky.

V grafe 7.2 je vidieť, že časová náročnosť narastá so zvyšujúcim sa stupňom derivácie. Hodnoty grafu boli získané spriemerovaním nameraných hodnôt pre jednotlivé derivácie vo všetkých štyroch ročných obdobiach. V priemere táto operácia trvá 75 ms.

Ďalšou náročnou operáciou bola grafická interpretácia vygenerovaných reťazcov. Namerané výsledky sú zhrnuté v grafe 7.2. Na obrázku je vidno, že čas potrebný na grafickú interpretáciu je mnohonásobne väčší než čas potrebný na generovanie ďalšej derivácie systému. Tento čas narastá najmä u derivácií s vyšším stupňom a pri dostatočnom množstve živín v prostredí. V takomto prípade dochádza k výraznému vývoju väčšiny stromov. To má za následok pridanie veľkého množstva nových grafických objektov do zobrazovanej scény. Treba podotknúť, že na vykresľovanie sú použité len základné grafické primitíva a modelovaná je len kostra stromu. Pridanie ďalších častí ako sú kvety, či listy by mohli túto operáciu ešte predĺžiť, nakoľko by to viedlo k vytváraniu ďalších objektov. V priemere táto operácia zaberie 155 ms, čo je viac než dvojnásobok oproti operácii generovania reťazca.



Obr. 7.2: Porovnanie generovania derivačného kroku s časom potrebným na vykreslenie objektov v scéne.

7.4 Zhrnutie

Testovaním bolo zistené, že navrhnutý systém L gramatík je schopný reagovať na podnety z prostredia. Prostredníctvom komunikácie zavedenej v otvorených L systémoch je prostredie schopné vplývať na vlastnosti zložkových L gramatík, ktoré tvoria spomínaný systém. Systém sa tak hodí na simulácie spojené so sledovaním prírodných javov ako je čerpanie živín z pôdy, či fotosyntéza.

Pri implementácii je vhodné zamerať sa na dve hlavné operácie, ktoré by mohli mať hlavný vplyv na celkovú výkonnosť programu. Prvou je samotné generovanie reťazca. Táto operácia závisí od komplexnosti generovaných reťazcov a teda úzko súvisí s použitými L gramatikami. Výpočtovo náročnejšia bola operácia grafickej interpretácie. Táto operácia trvala v priemere dvakrát dlhšie než generovanie reťazca. Budúci vývoj projektu by sa tak mal zamerať aj na grafické optimalizácie, týkajúce sa počtu, či spôsobu vykresľovania objektov v scéne.

Kapitola 8

Ďalší vývoj projektu

Táto kapitola sa zaoberá možným budúcim vývojom práce. Opísané sú možnosti rozšírenia, jednak použitého modelu stromu ako aj modelu prostredia. Takisto sú opísané pokročilé techniky, ktoré vedú k realistickejšiemu zobrazovaniu vykresľovaných grafických objektov.

8.1 Model stromu

Je známe, že pravidlá L gramatík sú schopné definovať nielen vývoj, či topológiu objektov ale taktiež ich správanie a animáciu v reálnom čase. Simulovaniu správania a animácii sa venujú páni H. Noser a P. Thalmann vo svojej práci [24].

Nami navrhnutý systém môže byť v budúcnosti rozšírený o ďalšie komplexnejšie modely stromov. Tým by sa dosiahlo ešte realistickejšie chovanie. Definovanie a vytvorenie takto komplexných pravidiel vyžaduje znalosti z oblasti teoretickej informatiky ako aj z oblasti botaniky, či molekulárnej biológie. Nakoľko je návrh takéhoto modelu zložitý, je možné v budúcnosti využiť framework, ktorý bol vyvinutý práve pre tieto účely. Trojica pánov z univerzity v Tongji vyvinula framework, na modelovanie stromov za pomoci L gramatík. Výhodou ich frameworku je, že stačí zadefinovať vstupné parametre pre daný model a vygenerovaná bude odpovedajúca L gramatika. Ide teda o rovnaký spôsob získania modelu s tým rozdielom, že zohľadnené sú viaceré biologické aspekty rastliny. Viac o frameworku ako aj možné vstupné parametre je možné nájsť v článku [33].

V prípade, ak by sa práca uberala smerom vylepšovania už implementovaného modelu, mohli by sa v modeli zohľadniť ďalšie parametre. Jedným z nich je pridanie tzv. *fylotaktického uhlu* z *angl. Phyllotactic angle*. Pojem *Phyllotaxis* resp. *Phyllotaxy* (zo starovekej gréčtiny *phýllon* = „list“ a *táxis* = „usporiadanie“) vyjadruje usporiadanie listov na stonke rastliny. Fylotaktický uhol vyjadruje uhol medzi rovinou dcérskych vetiev (listami) a rodičovskou vetvou (stonkou). Pridaním tohto parametru, by sa dosiahla väčšia podobnosť vykresľovaného modelu voči reálnemu stromu. Na druhej strane, by to viedlo k ďalším výpočtom, čo by mohlo mať negatívny dopad na plynulosť simulácie. Detailne sa tejto téme venuje kniha [15].

8.2 Model prostredia

Model prostredia poskytuje radu možností na vylepšenie. Je možné aplikáciu rozšíriť o viaceré zdroje živín alebo zakomponovať do výpočtu množstva odobratých živín zo zdroja aj

vzdialenosť konkrétneho stromu. To by viedlo k pomalšiemu vývinu vzdialenejších organizmov.

Ďalšou možnosťou by mohlo byť pridanie zložitejších procesov látkovej premeny, ktoré by mali vplyv na vývoj ako je napr. fotosyntéza. *Fotosyntéza* je biochemický proces zachytávania energie slnečného žiarenia a je využitá na fixáciu oxidu uhličitého v zelených rastlinách a niektorých prokaryotoch za vzniku karbohydrátov (sacharidov). Dochádza teda k premene prijatej energie svetelného žiarenia na energiu chemickej väzby pričom vznikajú organické látky z anorganických. Mitochondriálnym dýchaním môže byť takto do atmosféry späť uvoľnených až 70% CO_2, ktorý je fixovaný počas fotosyntézy [2]. Tento proces látkovej premeny prebieha v zelených častiach rastlín ako sú listy, zelené časti stonky, či púčiky kvetov. Do simulácie by sa tak pridalo simulovanie dňa a noci. Intenzita svetla by bola daná fázou dňa a aktuálnym ročným obdobím. Na základe toho by potom mohla byť ovplyvňovaná farba prípadne počet listov. Simulovať by sa tak dali javy ako opadávanie listov v jeseni alebo kvitnutie stromov na jar.

8.3 Grafické vylepšenia

Po grafickej stránke poskytuje implementovaná aplikácia radu možností na vylepšenie z ktorých niektoré sú uvedené v nasledujúcich častiach. Ide napríklad o lepší spôsob vykresľovania použitím kriviek. Pre väčšiu optimalizáciu vykresľovania by sa mohlo pristúpiť k vylepšeniu prepisovacích pravidiel. Posledným uvedeným rozšírením môže byť oblasť virtuálnej reality, ktorej integrácia do už implementovanej aplikácie, nemusí byť až taká zložitá.

8.3.1 Reprezentácia krivkami

Generovanie kriviek tvorí základ *CAD systémov* z *angl. Computer Aided Design*. Ich rozvoj bol stimulovaný potrebami strojárenského priemyslu. Práve vývoj CAD systémov viedol k rýchlemu vývoju počítačovej grafiky. Hlavné myšlienky sú z 50. až 60. rokov 20. storočia a možno povedať, že podstatným spôsobom zmenili prácu konštruktérov a návrhárov. S krivkami sa stretneme i v geografických a informačných systémoch (GIS), či pri generovaní fontov písiem textových editorov. Okrem parametrického vyjadrenia kriviek, existujú aj iné matematické popisy ako *bezériové* alebo *NURBS* krivky [31]. Použitím kriviek sme schopní popísať všetky základné geometrické objekty spolu s pokročilejšími tvarmi. Modelované tak môžu byť listy, kvety alebo prepracovanejšie časti kmeňa a vetiev.

8.3.2 Virtuálna realita

Virtuálna realita je technológia, ktorá sa v priebehu 21.storočia stala prístupná bežným ľuďom. Najväčším spúšťačom bol Google Cardboard[1], ktorý pomocou spojenia lacného kartónu a displeja smartfónu dokázal ľudí priviesť do 3D prostredia. Medzi základne vlastnosti virtuálnej reality patrí:

- Všetky deje sa uskutočňujú v reálnom čase, pokiaľ možno s okamžitou odozvou na akcie užívateľa.
- Virtuálny svet a objekty v ňom umiestnené majú trojrozmerný charakter, alebo aspoň vytvárajú jeho dojem.

[1] https://vr.google.com/cardboard/

- Užívateľ môže vstupovať do virtuálneho sveta a pohybovať sa v ňom po rozličných dráhach (chodí, lieta, skáče).
- Virtuálny svet nie je statický. S jeho časťami môže užívateľ manipulovať. Virtuálne telesá sa pohybujú po animačných krivkách, ovplyvňujú užívateľa a aj seba navzájom.

Virtuálnu realitu je tak možné použiť na prieskum a manipuláciu s dátami takým spôsobom ako to nebolo v minulosti možné [32]. Pridaním virtuálne reality by sa tak mohla vytvoriť virtuálna prehliadka lesa, resp. určitej časti ekosystému. Takéto rozšírenie programu je možné kvôli podpore virtuálnej reality v knižnici *BabylonJS* [29].

8.3.3 Závislosť prepisovacích pravidiel na natočení kamery

Pri pohľade na scénu je kamera natočená určitým smerom. Užívateľ tak vidí len určitú časť scény. V aplikácii je použitá kamera z pohľadu prvej osoby. Pri takomto pohľade, je zbytočné generovať príliš veľké množstvo detailov na odvrátenej strane. Riešením by mohlo byť zavedenie závislosti prepisovacích pravidiel na natočení kamery. L gramatiky tvoriace systém by tak mali vedomosť o pozícii kamery, čo by sa odzrkadlilo v scéne na množstve vykreslených detailov jednotlivých stromov.

Kapitola 9

Záver

Kapitola 2 poskytuje čitateľovi úvod do problematiky. V tejto časti bol uvedený prehľad základných pojmov a definícií spojených s bezkontextovými gramatikami. Tie patria medzi gramatiky s frázovou štruktúrou. Ich definícia bola taktiež súčasťou tohto úseku práce. Definované boli základné pojmy ako relácia priamej derivácie, či jazyk generovaný gramatikou.

Táto diplomová práca detailne popisovala problematiku gramatických systémov. V kapitole 3 bolo spomenuté ich základné delenie. Následne sme sa zamerali na paralelné gramatické systémy. Tie boli formálne zadefinované. Vysvetlené boli jednotlivé časti paralelných gramatických systémov. Každá komponenta systému mala vlastnú vetnú formu. V rámci každej časovej jednotky, každá komponenta aplikovala pravidlo a prepísala svoju vlastnú vetnú formu. Kľúčovým prvkom týchto gramatických systémov bol mechanizmus komunikácie pomocou dotazov. Tá sa uskutočňovala vďaka dotazovacím symbolom. Výhodou týchto systémov bolo, že na komunikáciu medzi jednotlivými komponentami, mohli byť použité aj iné spôsoby. Tento fakt bol ďalej využitý v časti s návrhom výslednému systému L gramatík, kde sa využil spôsob komunikácie známy z otvorených L gramatík.

Ďalej boli rozobraté rôzne druhy L gramatík spolu so spôsobmi ich využitia pri modelovaní rastlinných organizmov. Porovnané boli bezkontextové gramatiky a L gramatiky. Z oblasti L gramatík sa práca zameriavala najmä na modelovanie vývoja biologických organizmov v čase. Na celú problematiku bolo nahliadané z pohľadu simulácie reálneho chovania. Popísaný bol základný druh L gramatík, d0L gramatiky. Ten sa stal teoretickým základom pre zložitejšie typy L gramatík. Išlo predovšetkým o stochastické L gramatiky, parametrické L gramatiky a L gramatiky s kontextovou podmienkou. Popísané boli taktiež otvorené L gramatiky, u ktorých dochádza k výmene informácií medzi L gramatikou a prostredím. Komunikácia prebieha obidvoma smermi, pričom prostredie informuje rastlinu o aktuálnom množstve živín. Na druhej strane informuje rastlina prostredie o množstve odobratých živín z prostredia. Na tomto koncepte bola založená aj implementovaná aplikácia.

Získané teoretické poznatky boli využité pri návrhu finálneho riešenia. Navrhnutý bol systém L gramatík. Ten pozostával z viacerých otvorených L gramatík, ktoré komunikujú s prostredím. Tento systém bol najprv formálne zadefinovaný. Formálne definície systému obsahuje kapitola 5. Okrem toho bola táto kapitola doplnená o formálnu definíciu otvorených L gramatík. Tá bola odvodená na základe tvrdení uvedených v časti 4. Navrhnutý a formálne zadefinovaný tak bol paralelný systém otvorených L gramatík. Uvedené formalizmy boli implementované.

Navrhnutá bola webová aplikácia. Na implementáciu bol použitý programovací jazyk *Javascript* v kombinácii s knižnicou na 3D vykresľovanie *BabylonJS*. Užívateľské rozhranie bolo vytvorené za pomoci knižnice *React*. Pri implementácii bol kladený dôraz na použi-

tie modelov z reálneho sveta. Implementovaný bol druh stromu z *lat. Alstonia Scholaris*. Tento strom pochádza z oblasti južnej a juhovýchodnej Ázie. Výsledná simulácia sa tak snaží simulovať podmienky typické práve pre túto oblasť sveta. Simulovaný je vývoj uvedeného druhu stromu pri rôznych ročných obdobiach. Jednotlivé modely stromov odoberajú z prostredia živiny. Tie sú do prostredia dopĺňané na základe údajov získaných z meteorologických meraní pre oblasť Pekingu. Prostredie tak priamo ovplyvňuje stupeň vývoja stromov, pričom parametre modelov sú počítané v reálnom čase. Pozorovať je tak možné vývoj celej skupiny rastlinných organizmov, ktoré spolu tvoria fungujúci systém.

Okrem detailného popisu zaujímavých častí implementácie, práca obsahuje radu schém a ukážok pre lepšie preniknutie do problematiky. Popísané boli dátové štruktúry a celý koncept vnútornej logiky. Nakoľko aplikácia disponuje užívateľským rozhraním, bola časť práce venovaná vstupným parametrom simulácie, ktoré môže pred spustením užívateľ zadať. Okrem toho bolo v skratke popísané ovládanie simulácie a význam zobrazovaných údajov.

Testovaniu a skúmaniu vlastností implementovaného programu bola venovaná kapitola 7. Aplikácia bola testovaná z pohľadu simulácie a výkonnosti. Pri testovaní simulácie rôznych podmienok v rámci jednotlivých ročných období bolo ukázané, že navrhnutý systém L gramatík je schopný reagovať na zmeny v prostredí. Obojsmerný tok informácii v systéme nám umožňuje simulovať pokročilejšie prírodné javy. Rôznorodosť prenášaných parametrov nám ďalej poskytuje možnosti rozšírenia systému. Prenášané tak môžu byť ďalšie dôležité informácie a pozorované zložitejšie prírodné deje. Samozrejme pre správne definovanie chovania sú nutné pokročilé znalosti z oblasti botaniky ako aj teoretickej informatiky. Výkonnostné testy boli cielené na dobu generovania reťazca a čas potrebný na jeho grafickú interpretáciu. Z nameraných údajov vyplynulo, že časovo náročnejšia bola operácia interpretácie. V priemere táto operácia trvala dvakrát dlhšie než generovanie reťazca v rovnakom derivačnom kroku. Práve rýchlosť generovania reťazcov ako aj ich grafická vizualizácia mali hlavný vplyv na plynulosť vykresľovania objektov.

Aplikácia v súčasnom stave poskytuje priestor pre ďalší vývoj. Ide najmä o optimalizácie vykresľovania, či zvýšenie rýchlosti generovania ďalších derivácií. Takisto je možné doplniť pokročilejšie vizuálne prvky ako tiene alebo iné materiály a poskytnúť tak užívateľovi lepší estetický zážitok. V neposlednom rade je možné aplikáciu rozšíriť o ďalšie modely stromov alebo iné podmienky v ktorom sa dané organizmy vyskytujú. Modely stromov by mohli byť komplexnejšie a popisovať tak ešte detailnejšie prírodné procesy, ktoré prebiehajú v životnom prostredí. Simulácia by tak bola ešte reálnejšia. Taktiež je možné aplikáciu doplniť o pokročilejšie animácie vývinu jednotlivých častí použitého modelu. Možnosti ďalšieho vývoja projektu boli opísané v kapitole 8.

Pre popis komplexných modelov a simulovanie pokročilých dejov výrazne narastá zložitosť pravidiel vstupnej L gramatiky. Výsledný reťazec je zložitý, čo značne predlžuje jeho rozgenerovanie v ďalších deriváciách. Obzvlášť, ak je potrebné rozgenerovať hneď niekoľko reťazcov pre viacero stromov tak, ako je to aj v našom prípade. V budúcnosti by teda bolo dobré zamerať sa na možnosti efektívneho generovania reťazca, či možností redukovania zložitosti prepisovacích pravidiel.

Literatúra

[1] Aho, A. V.; Lam, M. S.; Sethi, R.; aj.: *Compilers: principles, techniques & tools*. Pearson/Addison Wesley, druhé vydanie, 2007, ISBN 978-0321486813.

[2] Bacon, K.: *Photosynthesis : photobiochemistry and photobiophysics*. Dordrecht Boston: Kluwer Academic Publishers, 2001, ISBN 978-0-306-48136-9.

[3] Balasubramanian, A.: *Ecosystem and its components*. Technická správa, Centre for Advanced Studies in Earth Science, University of Mysore, Manasagangotri, Feb 2008.
URL https://www.researchgate.net/publication/314213426_ECOSYSTEM_AND_ITS_COMPONENTS

[4] Borchert, R.; Honda, H.: *Control of Development in the Bifurcating Branch System of Tabebuia rosea: A Computer Simulation. International Journal of Plant Sciences*, ročník 145, č. 2, Jun 1984.

[5] Catuhe, D.; Rousset, D.; Vandenberghe, S.: *A complete JavaScript framework for building 3D games and experiences with HTML5, WebGL, WebVR and Web Audio*. [Online; navštívené 09.04.2018].
URL https://www.babylonjs.com/

[6] Chyr, W.: *32 L-Systems*. Jan 2012, [Online; navštívené 10.12.2017].
URL http://williamchyr.com/tag/generative-art/

[7] Dalton, J. D.: *A modern JavaScript utility library delivering modularity, performance and extras*. [Online; navštívené 09.04.2018].
URL https://lodash.com/

[8] Deussen, O.; Hanrahan, P.; Lintermann, B.; aj.: *Realistic modeling and rendering of plant ecosystems. Proceedings of the 25th annual conference on Computer graphics and interactive techniques - SIGGRAPH 98*, 1998.

[9] Facebook Inc.: *Flow a static type checker for JavaScript*. 2014–2018, [Online; navštívené 09.04.2018].
URL https://flow.org/

[10] Facebook Inc.: *React a JavaScript library for building user interfaces*. 2018, [Online; navštívené 09.04.2018].
URL https://reactjs.org

[11] Gardner, S.; Sidisunthorn, P.; May, L. E.: *Heritage Trees of Penang*. Areca Books, 2011, ISBN 9675719060.

[12] Hanan, J. S.: *Parametric L-systems and their application to the modelling and visualization of plants*. Dizertačná práca, University of Regina, Jun 1992.

[13] Holiday Weather: *Weather and temperature averages for Beijing, China*. 2018, [Online; navštívené 09.04.2018].
URL http://www.holiday-weather.com/beijing/averages/

[14] JavasSript.com: *What is JavaScript*. 2016, [Online; navštívené 09.04.2018].
URL https://www.javascript.com/

[15] Juan, R.: *Growth Patterns in Physical Sciences and Biology*. Boston, MA: Springer US Imprint Springer, 1993, ISBN 978-1-4613-6235-7.

[16] Jungck, J.; Spangenberg, J.; Khiripet, N.; aj.: *3D FractaL-Tree*. [Online; navštívené 09.04.2018].
URL
http://bioquest.org/esteem/esteem_details.php?product_id=403

[17] Koutný, J.: *L systémy a jejich aplikace*. Diplomová práca, Vysoké učení technické v Brně, Fakulta informačních technologií, 2008.
URL https://www.vutbr.cz/www_base/
zav_prace_soubor_verejne.php?file_id=116311

[18] Lee, R. E.: *Phycology*. Cambridge University Press, Štvrté vydanie, 2008, ISBN 978-0-521-63883-8.

[19] Lindenmayer, A.: *Developmental algorithms: Lineage versus interactive control mechanisms. Bulletin of Mathematical Biology*, 1982: s. 219–245.

[20] Martin, J.: *Algorithmic Beauty of Buildings Methods for Procedural Building Generation*. Computer science honors theses, Trinity University, Mar 2005.

[21] Meduna, A.: *Formal Languages and Computation: Models and Their Applications*. Auerbach, prvé vydanie, 2014, ISBN 978-1466513457.

[22] Meduna, A.; Švec, M.: *Grammars With Context Conditions and Their Applications*. John Wiley & Sons, prvé vydanie, 2005, ISBN 978-0471718314.

[23] Mitchison, G. J.; Wilcox, M.: *Rules governing cell division in Anabaena. Nature*, ročník 239, Sep 1972, ISSN 239:110–111.

[24] Noser, H.; Thalmann, D.: *A rule-based interactive behavioral animation system for humanoids. IEEE Transactions on Visualization and Computer Graphics*, ročník 5, č. 4, Oct 1999: s. 281–307, ISSN 1077-2626, doi:10.1109/2945.817347.

[25] Peringer, P.; Hrubý, M.: *Modelování a simulace*. Učebné materiály, 2017, [Online; navštívené 09.04.2018].
URL https://wis.fit.vutbr.cz/FIT/st/course-files-st.php?file=
%2Fcourse%2FIMS-IT%2Flectures%2FIMS.pdf&cid=10316

[26] Peterson, J. L.: *Petri Net Theory and the Modeling of Systems*. Prentice Hall, 1981, ISBN 0136619835.

[27] Prusinkiewicz, P.; Lindenmayer, A.: *The algorithmic beauty of plants*. Springer–Verlag, druhé vydanie, 1996, ISBN 978-0387972978.

[28] Prusinkiewicz, P.; Měch, R.: *Visual Models of Plants Interacting with Their Environment. Proceedings of SIGGRAPH 96*, Aug 1996: s. 397–410.

[29] Rousset, D.: *From zero to hero, creating WebVR experiences with Babylon.js on all platforms*. Jul 2017, [Online; navštívené 09.04.2018].
URL https://www.davrous.com/2017/07/07/from-zero-to-hero-creating-webvr-experiences-with-babylon-js-on-all-platforms/

[30] Rozenberg, G.; Salomaa, A.: *Handbook of Formal Languages: Volume 2. Linear Modeling: Background and Application*, ročník 2. Springer, prvé vydanie, 2011, ISBN 978-3642082306.

[31] Španěl, M.: *Křivky v počítačové grafice*. Učebné materiály, 2016, [Online; navštívené 09.04.2018].
URL https://www.fit.vutbr.cz/study/courses/IZG/private/lecture/izg_slide_krivky_print.pdf

[32] Šubák, M.: *Čo je to virtuálna realita - VirtualnaRealita.eu*. 2017, [Online; navštívené 09.04.2018].
URL https://virtualnarealita.eu/co-je-virtualna-realita/

[33] Sun, R.; Jia, J.; Jaeger, M.: *Intelligent tree modeling based on L-system*. In *2009 IEEE 10th International Conference on Computer-Aided Industrial Design Conceptual Design*, Nov 2009, s. 1096–1100, doi:10.1109/CAIDCD.2009.5375256.

[34] The Khronos Group Inc.: *OpenGL ES for the Web*. [Online; navštívené 09.04.2018].
URL https://www.khronos.org/webgl/

[35] Togelius, J.; Shaker, N.; Dormans, J.: *Grammars and L-systems with applications to vegetation and levels. Procedural Content Generation in Games Computational Synthesis and Creative Systems*, 2016: s. 73–98, [Online; navštívené 08.12.2017].
URL http://pcgbook.com/wp-content/uploads/chapter05.pdf

[36] USDA, Agricultural Research Service, National Plant Germplasm System: *Germplasm Resources Information Network (GRIN-Taxonomy)*. 2018, [Online; navštívené 09.4.2018].
URL https://npgsweb.ars-grin.gov/gringlobal/taxonomydetail.aspx?id=2688

[37] Viruchpinta, R.; Khiripet, N.: *Real-time 3D Plant Structure Modeling by L-System with Actual Measurement Parameters*. Technická správa, National Electronics and Computer Technology Center, Pathumthani 12120, Thailand, 2011.
URL https://www.researchgate.net/publication/268380530_Real-time_3D_Plant_Structure_Modeling_by_L-System_with_Actual_Measurement_Parameters

Zoznam obrázkov

Zoznam tabuliek

Príloha A

Obsah priloženého pamäťového média

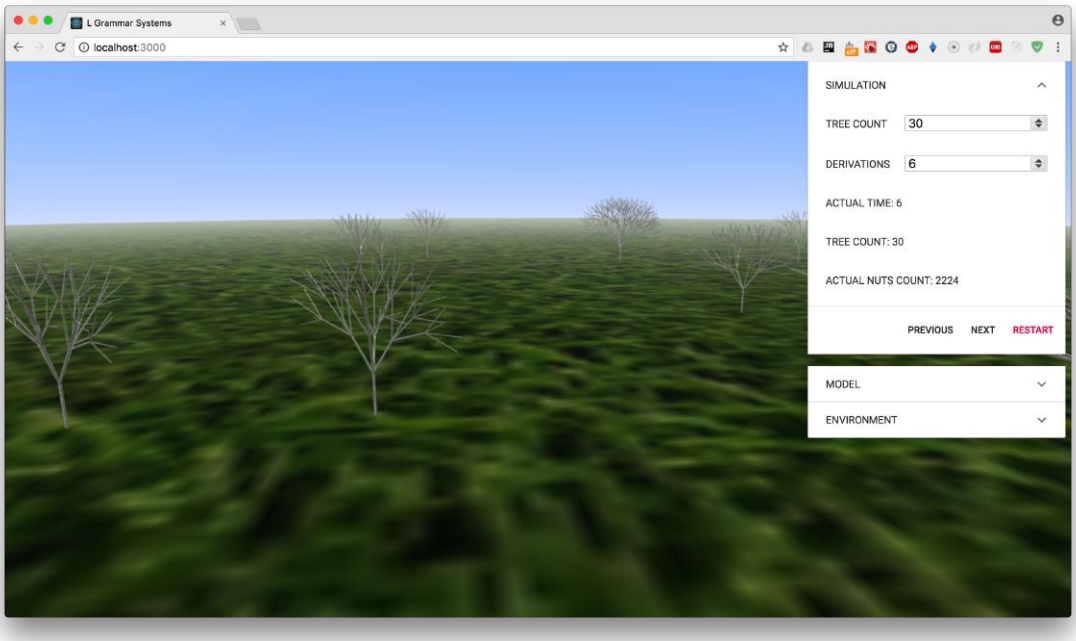
- */src/* – zdrojové súbory programu
- */doc/* – zdrojové súbory technickej správy
- *xhnatp00-Systemy-L-gramatik.pdf* – technická správa
- *README.md* – manuál k inštalácii a spusteniu programu

Príloha B

Užívateľské rozhranie

Aplikácia disponuje jednoduchým užívateľským rozhraním. To je zobrazené v pravej časti obrazovky tak ako je uvedené na obrázku B.1. Menu je rozdelené na tri časti, pričom každá z nich obsahuje nastaviteľné položky prípadne zobrazuje ďalšie informácie:

- Časť SIMULATION – nastavenia simulácie.
  - TREE COUNT – požadovaný počet generovaných stromov. Skutočný počet stromov v scéne sa však môže líšiť v závislosti od počtu živín v prostredí.
  - DERIVATIONS – počet požadovaných derivácií systému. V prípade krokovania sa mení aktuálny čas v položke ACTUAL TIME, ktorý odpovedá poradovému číslu derivácie.
  - ACTUAL TIME – aktuálny simulačný čas. Pri krokovaní dochádza k jeho inkrementácii resp. dekrementácii.
  - ACTUAL NUTS COUNT – aktuálny počet vygenerovaných živín v prostredí. Tie sú následne prerozdelené medzi stromy.
- Ovládacie prvky simulácie:
  - RUN – spustenie simulácie.
  - PAUSE – pozastavenie simulácie v aktuálnom čase. V tomto režime je potom možné krokovanie simulácie prípadne reštartovanie.
  - RESTART – reštartuje simuláciu, zmaže vykreslené objekty zo scény.
  - NEXT – prevedie nasledujúci derivačný krok v závislosti od aktuálnej derivácie (simulačný čas = derivácia).
  - PREVIOUS – vykreslí predchádzajúci derivačný krok v závislosti od aktuálnej derivácie (simulačný čas = derivácia).
- Časť MODEL – nastavenia rastlinného modelu.
  - SPLIT ANGLE – uhol vetvenia.
- Časť ENVIRONMENT – nastavenia prostredia.
  - NUTRIENTS – počiatočné množstvo živín v prostredí.
  - WEATHER SEASON – zvolené ročné obdobie. Vplýva na množstvo živín, ktoré je generované do prostredia.

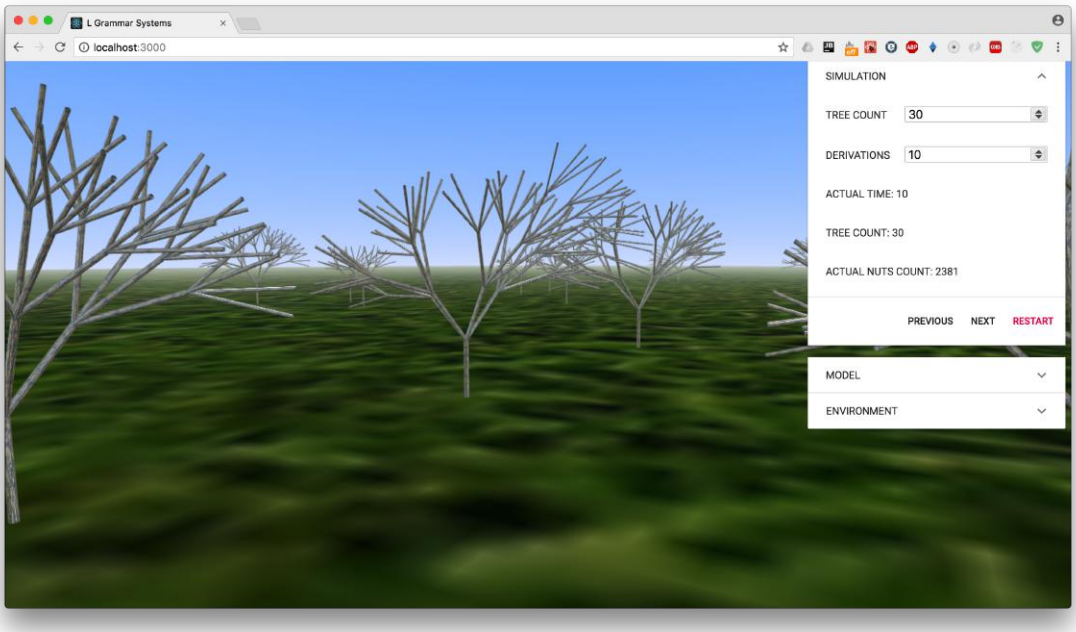


Obr. B.1: Užívateľské rozhranie implementovanej aplikácie.

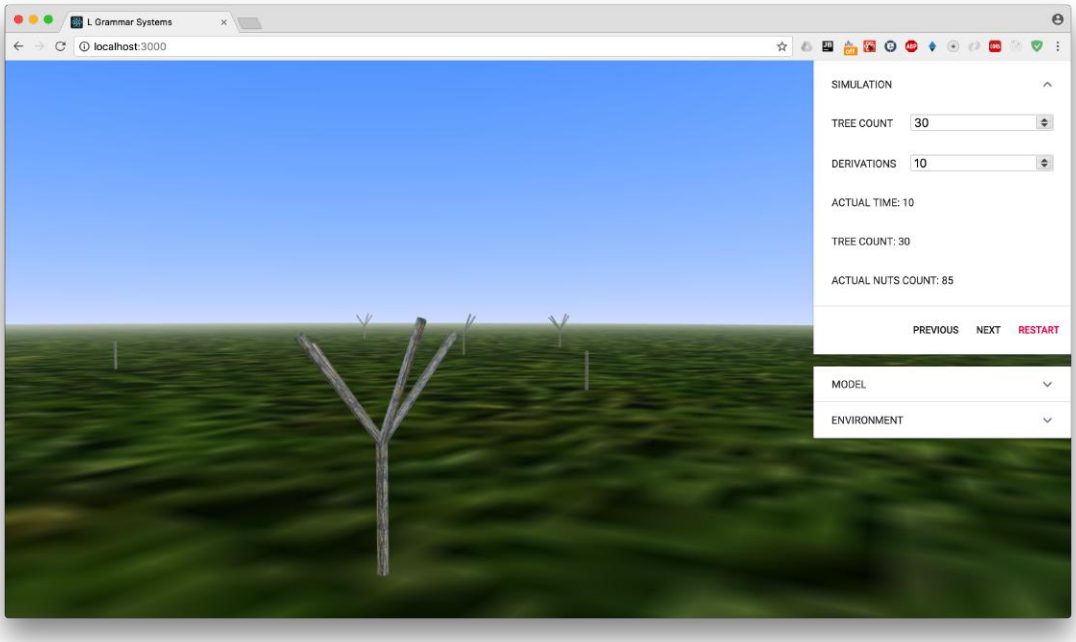
Príloha C

Grafický výstup z aplikácie

Táto príloha obsahuje grafický výstup z aplikácie. Obrázky C.1 a C.2 ilustrujú vývoj ekosystému pri rovnakých parametroch a rozličných ročných obdobiach. Uvedené sú výstupy pre zimu a leto. Práve pri týchto ročných obdobiach je rozdiel v množstve vygenerovaných grafických primitív badateľný. Už na prvý pohľad je vidieť, že vývoj stromov v lete je omnoho väčši než v zime. Uvedené obrázky sú vygenerované pre desiaty derivačný krok a rovnaké parametre, ako sú uvedené v podkapitole 7.2. Pre lepšie objasnenie však treba uviesť, že pozícia stromov ako aj prísun živín do prostredia je generovaný náhodne. Pri opätovnom spúšťaní programu tak dostávame mierne odlišné výsledky.



Obr. C.1: Vygenerovaný ekosystém pre letné obdobie.



Obr. C.2: Vygenerovaný ekosystém pre zimné obdobie.

Register